Arix marketplace lending

Arix is the automated loan validation and funding software from Cross River Bank (CRB). As a marketplace lending (MPL) partner with CRB, you can access Arix functions via an extensive list of API calls.

The scenarios and diagrams in this guide illustrate the funding flow for loans. Loans are funded after the review and fulfillment of documentation and compliance requirements.

>
>
> ### Arix status page
>
> You can sign up for maintenance updates and see the current status of the Arix system at **https://crossriverarix.statuspage.io/**

Click here for our blogs.

Loan process with no failures

Loan process with no failures

In the typical scenario, Arix receives a loan request and moves the loan forward to approval and funding because there aren't any outstanding issues.

- Partner sends the loan payload to CRB.
- Arix assigns a GUID to the loan.
  Status: Received
- Partner sends all the required loan documents to Arix.
- Arix automatically reviews and confirms that the documents constitute a complete set.
  Status: Docs Complete
- Arix automatically performs the required compliance checks.
  Status: Passed Compliance
- Arix automatically approves the loan.
  Status: Approved
- Arix passes the loan through to funding.
  Status: InFunding
- Arix funds the loan.
  Status: Funded

Detailed steps to a loan with no failures

Partner actions:

1. Send CRB a signed loan agreement.
   Arix assigns a GUID to the loan and marks it **Received**.
2. When the loan is marked **Received**, send all the required loan documents to Arix.

Arix actions:

1. Arix automatically checks that the documents constitute a complete set.
   (An agreed upon document set is established at the beginning of the relationship with CRB.)
2. When all required documents are received, Arix marks the loan as **Docs Complete**.
3. Arix automatically runs the compliance rules.
4. When the loan passes compliance, Arix marks the loan as **Passed Compliance**.
5. The loan is then automatically moved into **Approved**.
6. The loan is funded.

Loan process with document failure

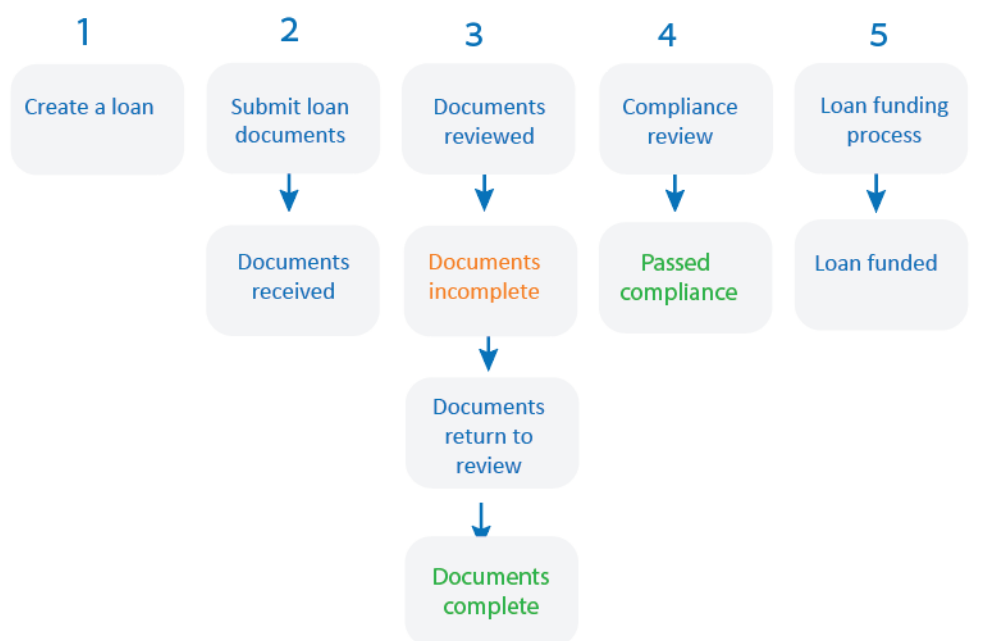
In this scenario, Arix received the loan request, but the loan can't be funded due to an incomplete document set.

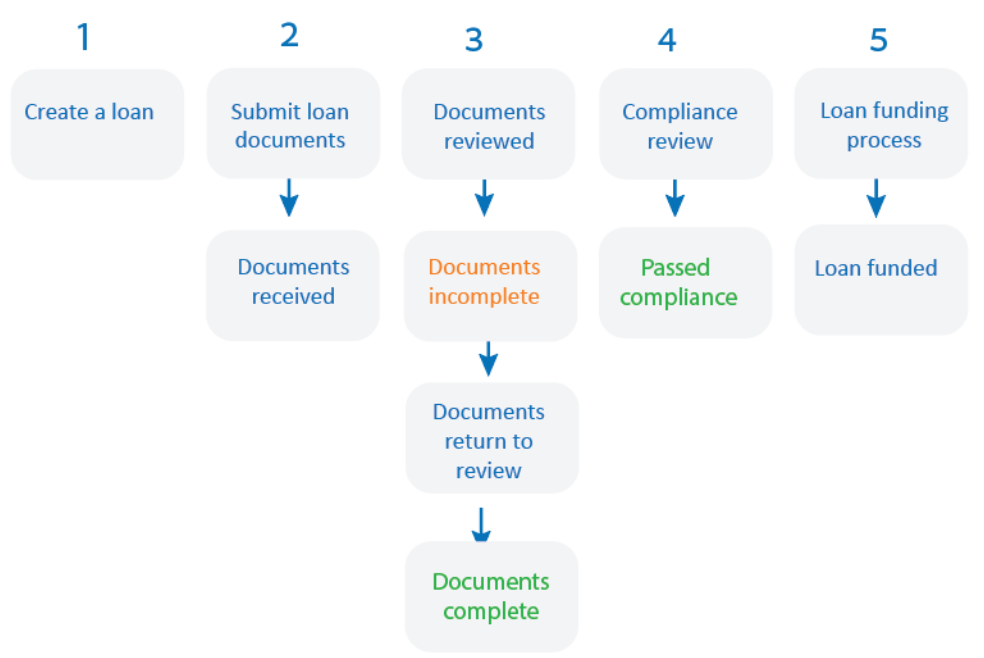
- Partner sends the loan payload to CRB.
- Arix assigns a GUID to the loan.
  Status: Received
- Partner sends the loan documents to CRB.
- Arix automatically checks that the documents constitute a complete set.

> ❗
>
> ### Important
>
> If all required documents are not sent, the status of the loan remains **Received** until the documents are complete. **No webhook is sent to notify that docs are incomplete**.

Status: Received

- When the document set is complete, the status changes.
  Status: Docs Complete
- Arix automatically performs the required compliance checks.
  Status: Passed Compliance
- Arix automatically approves the loan.
  Status: Approved
- Arix passes the loan through to funding.
  Status: InFunding
- Arix funds the loan.
  Status: Funded





Detailed steps for a loan with documentation failures

Partner actions:

1. Send CRB a signed loan agreement.
   Arix assigns a GUID to the loan and marks it **Received**.
2. When the loan is marked **Received**, send all the required loan documents to Arix.
3. If the loan remains in **Received**, resubmit the missing or corrected documents to Arix. You can see what was sent and what is required by calling Get loan details.

Arix actions:

1. Arix automatically checks that the documents constitute a complete set.
   (An agreed upon documentation set is established at the beginning of your relationship with CRB.)
   Make sure to check that the documents are complete. If they aren't complete, Arix keeps the loan in the **Received** state.
2. When the required documents are received, Arix marks the loan as **Docs Complete**.
3. Arix automatically runs the compliance rules.
4. When the loan passes compliance, Arix marks the loan as **Passed Compliance**.
5. The loan is then automatically moved into **Approved**.
6. The loan is funded.

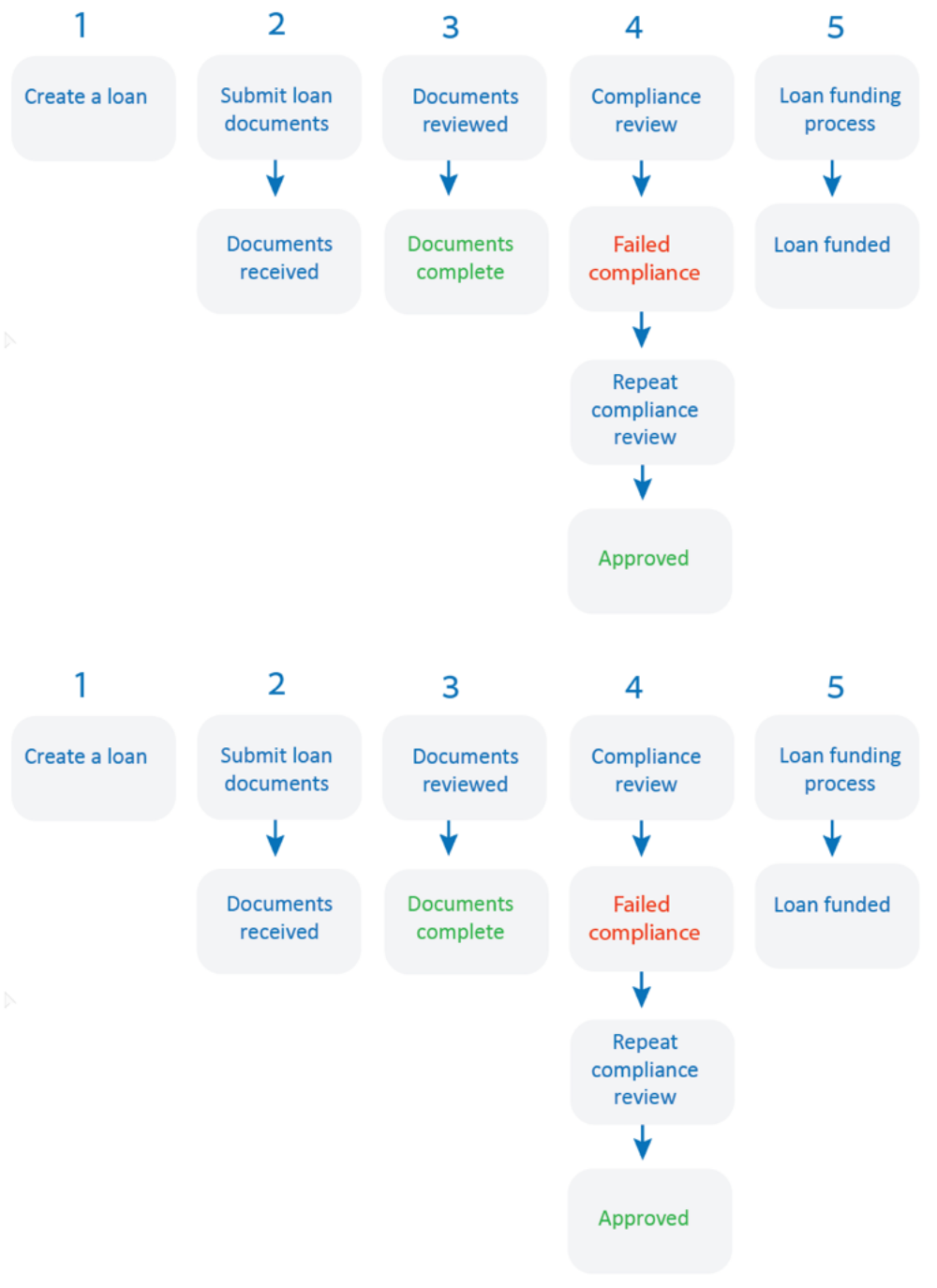
Loan process with compliance failure

In this scenario, Arix received the loan request, the documents were checked and found complete, but the loan can't move forward due to a compliance failure.

- Partner sends the loan payload to CRB.
- Arix assigns a GUID to the loan.
  Status: Received
- Partner sends the required loan documents to CRB.
- Arix automatically checks that the documents constitute a complete set.
  Status: Docs Complete
- Arix automatically runs the compliance rules and at least one rule fails.

For examples of a loan failure, go to compliance.

During the implementation period, we work with you in the sandbox to tune the loan payload, documentation, and rules. Our goal is that 100% of the loans pass compliance automatically without any failures.



Detailed steps for a loan with compliance failures

Partner actions:
The partner reviews the compliance failures and resubmits updates via the **LoanUpdate** API or reposts attachments, as applicable. Arix then performs the next steps toward loan approval.

Arix actions:
If the loan doesn't pass compliance for any reason, this can happen:

- The failure is resolved internally by the CRB Ops team.
- CRB Ops communicates with your operations team and confirms the additional information that is needed to approve the loan.

Note: You can receive updates on the compliance status with webhooks. Click here to view the webhooks available to register to.

Loan funding process

The Arix funding engine allows you the flexibility to control how the loan is funded. You can enter instructions and control the rails and the order in which to attempt to fund over those rails. You can also elect to fund different parts of the loan over different rails and supply fallback options if rails fail.

After the loan is approved for funding, Arix initiates funding. The loan passes to the **InFunding** status and funds are sent using rails.
Note: Any loan update made via a /**PUT** call is ignored when the loan is **InFunding**.

There are several rail types you can use, click here for more information. Depending on the rail type, funding results might be immediate or take up to a few days.

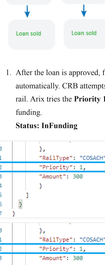
After funding, the loan passes through these stages:

- The funds are sent and received successfully
- The loan seasoning period begins
- The loan is sold

Arix generates webhooks which keep you up to date on the loan status.

Based on how the funding progresses, the loan can be in one of these statuses:

- **InFunding** *leading to* **Funded**
- **NotFullyFunded**
- **Returned**



1. After the loan is approved, funding starts automatically. CRB attempts to process the first rail. Arix tries the **Priority 1** rail first to start funding.
   Status: InFunding

```
        },
        "RailType": "COSACH",
        "Priority": 1,
        "Amount": 300
      }
    ]
}
```

```
        },
        "RailType": "COSACH",
        "Priority": 1,
        "Amount": 300
      }
    ]
}
```

If all the rails are assigned the same priority, Arix works through the loans in the order they were submitted, such as priority 2, priority 3, and so on. If no amount is specified in the request, the funding engine attempts to fund the full loan amount.

2. When all the rails are successfully funded, the status of the loan changes to funded.
   Status: Funded
3. After the loan has funded, the loan seasoning period begins, which typically occurs on business day 2 or 3 from the funding date.

In most cases the funding goes smoothly and the funds are fully received by the borrower. However, funding requests can fail immediately or succeed, and then be returned later.

If the loan doesn't fully fund from the first rail (priority 1), the engine looks for the priority 2 rail. A priority 2 rail can be:

- A rail that was originally designated as a priority 2 (to follow a priority 1 rail)
- The next priority 1 rail

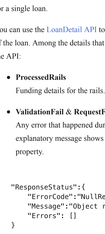
The funding engine continues to process rails until there are no rails left to process.

If after all the rails have been processed there is an outstanding loan balance, the loan status is then updated to **NotFullyFunded**. When **NotFullyFunded**, you can submit new funding information.

If the loan is fully funded prior to the last rail, any rails that weren't attempted are labeled **Skipped**. A message shows in the **LoanDetail** API explaining the reason for the skipped rail.

Failed, rejected, or returned rails

If a rail fails, is rejected, or returned, Arix reports a detailed error message as part of the **railUpdate** webhook and in the **LoanDetail** API.



Arix records all the funding attempts and continuously updates the status as each rail is processed, funded, or failed. If the loan wasn't fully funded and is in the status, **NotFullyFunded**, you can submit new funding information 2 more times. This is also the process for required multiple disbursements over a period of time for a single loan.

You can use the LoanDetail API to follow the progress of the loan. Among the details that are at the bottom of the API:

- **ProcessedRails**
  Funding details for the rails.

- **ValidationFail** & **RequestFailure**
  Any error that happened during rail funding. An explanatory message shows in the **Message** property.

```
{
  "ResponseStatus":{
      "ErrorCode":"NullReferenceException"
      "Message":"Object reference not set
      "Errors": []
  }
}
```

Note: If a loan was **Funded** and you have to make changes to the loan, you can send an Adjustment action via **PUT** /Loan/{Id}/FundingInfo.

Loan seasoning period

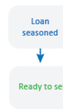
Every loan that is funded undergoes loan seasoning. The loan seasoning period typically occurs on business days 2 or 3.

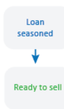
During this time funds are accepted by the receiving party. In some cases, funds are returned and the rails have to be resent.

Note: Every time funds are (initially) sent or resent, a new loan seasoning period starts. Loan seasoning terms and the loan season period are agreed upon by CRB and the partner. Arix is set for a default seasoning period of 3 business days.

Seasoning period ended and the loan successfully funded

At the end of seasoning period when receipt of the funds has been confirmed, the loan is ready to be sold.
Status: ReadyToSell



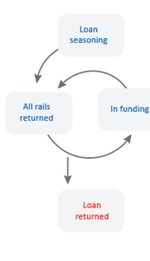


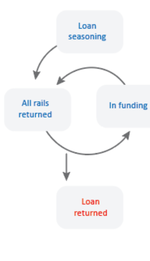
Loan is in seasoning, some rails are returned and the loan keeps trying to fund

- If some of the rails are returned during the seasoning period, the loan status changes to reflect that the loan is not fully funded.
  Status: NotFullyFunded
- Arix prompts you to resubmit funding information. Use PUT /Loan/{Id}/FundingInfo to update the loan funding info.
- The system waits for the loan to enter the **Funded** status again.
- The seasoning period restarts.

Seasoning period ends and rails are returned

If all rails are returned during this seasoning period, the loan status changes.
Status: Returned





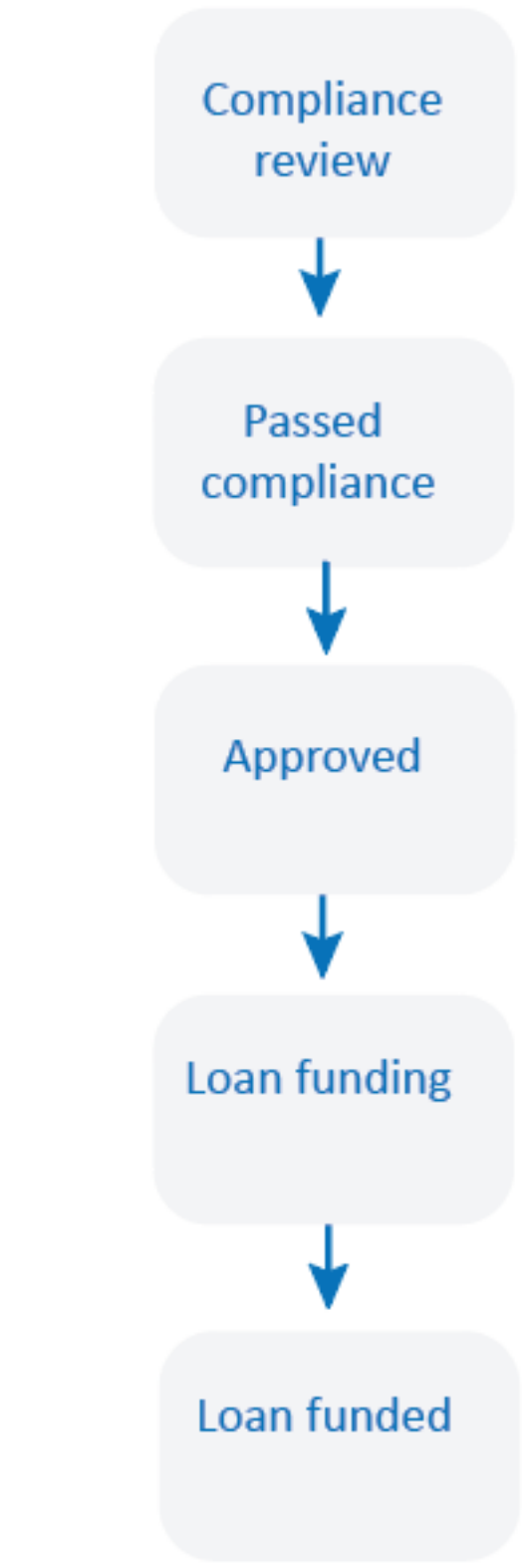
Compliance

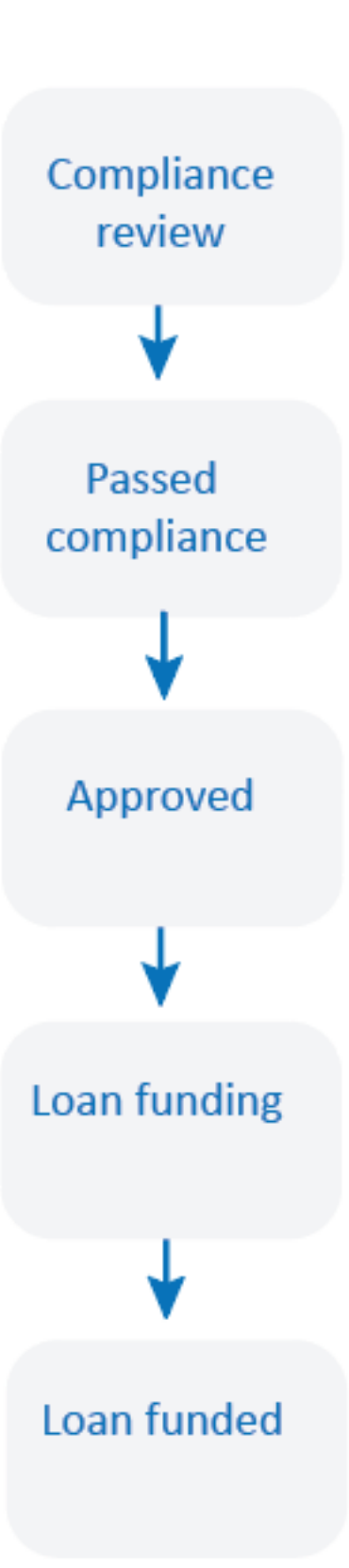
The compliance engine within Arix verifies that a loan adheres to your AML/BSA processes and Credit & Underwriting policy. This allows CRB to demonstrate to a regulator how it provides oversight on each and every loan. As part of the implementation process, we code the rules, which are fine tuned during the testing phase of the implementation.

If a loan meets all the compliance rules, it passes. If it fails, the status changes automatically to **ComplianceFailed**.

Detailed reasons for rule failures are available via the LoanDetail API and we recommend that you monitor those failures proactively. CRB Operations will reach out to your Operations department when we need additional information to clear a rule failure.

After Arix has verified that the loan is in compliance, it passes the loan to approval and then it gets funded.
Status: Passed Compliance





After you download the collection of Arix APIs, you can use these steps in the sandbox to practice failing loans based on compliance violations. The call will show the loan in the status, **ComplianceFailed**.

Steps for checking the status of a loan:

1. Create a loan request.
2. Add documentation to the loan request.
3. Update details in the loan or update the loan funding info.
4. Click **Send**.
5. Wait a few minutes, and then get the loan details with that ID.
6. Scroll down to the section **ComplianceRuleResultList** and check the compliance responses.

Compliance rule: Loan must be between $1,000 and $37,800 (inclusive)
Loan request: $500

The amount of the loan violates the rule and the loan fails. In the response message below, the `Result` shows `False` which indicates a compliance failure.

When a loan fails a compliance rule, the Operations team at CRB can see the failure and the related reason and communicate with you.

Use cases

Non-Revolving Lines of Credit

Twins, Amy and Dan Hansen, were finally off to college. After calculating the cost of college and consulting with their MPL, Mr. and Mrs. Hansen decided to apply for a Non-Revolving Line of Credit.

The credit line was approved and processed through Arix, and the funds were deposited into their account as a lump sum.

During a crisis, Janet accrued 1000's of dollars of credit card debt. To organize her debts, she contacted an MPL for a loan, in the hopes of consolidating her debts.

When she was approved, Janet opted to have Arix pay the creditors directly rather than have the funds deposited directly into her bank account.

Point-of-Sale (POS) loans

Sue and Jo were purchasing new mattresses, and, at checkout, were offered the option of **Point-of-Sale (POS)** financing. The clerk explained that instead of paying with cash or credit at checkout, they could apply for a loan right there. If approved, the purchase cost is passed into a loan, which they can pay out in installments. The MPL sets up monthly payment installments for their customers.

POS can be used for in-store or online purchases if the merchant is registered with the MPL and Arix. If approved in-store during checkout, an app on their phone shows a QR code which is passed to the clerk to complete the purchase. Arix sends payment to the vendor.

The Jacksons want to enlarge their kitchen but don't have enough money to pay it out fully. They're both employed however, and can pay monthly payments. They approach an MPL to apply for a loan.

The MPL creates a loan request in Arix on behalf of the Jacksons. They're approved and they have the MPL transfer the down payment directly to the contractor.

Each time the contractor asks for another incremental payment, the MPL sends a funding request to Arix, and Arix sends the funds directly to the contractor. Every payment is funded until the full amount has been paid out.

SBA (Small Business Administration) PPP Program loans

The Small Business Administration (SBA) offers emergency relief loans to small businesses, with CRB acting as the originator. The small business submits their loan paperwork to the MPL or CRB directly.

When a successful response is received from the SBA, Arix processes the application and funds the loan.

The *XYZ* textile factory offers their employees the option of taking small payroll loans. Several employees approached the payroll department and asked for an advance on their salary (a loan) because the holiday season was approaching.

After agreeing to the loan, the payroll department, via an MPL, used Arix to fund the loan directly to their employee's bank accounts.

Before undergoing a routine surgery, Alan decided to apply for a loan because he didn't want the extra pressure of paying out a large sum of money at one time.

Once approved for the loan via the MPL, the funds are transferred directly via Arix, to the medical service provider.

Homeowners Jack and Susan, were reviewing their electricity bills and noted that the bills kept increasing year after year. They decided to explore solar energy in an attempt to lower their energy bills, and as an added bonus, reduce their carbon footprint.

After being approved for the loan, Arix funded the vendor directly for each incremental payment until the full amount was paid out.

Sales flow

After the loan is funded, it is either sold back to the MPL, sold to an investor (directly), sold to the MPL on behalf of an investor (indirectly), or retained by CRB.

When the loan completes the seasoning period it is **Ready to Sell** to the MPL or an investor. CRB produces a pre-purchase report, which is an invoice.

Pre-purchase report (invoice)

The pre-purchase reports are sent to the MPL or investors the day before sale. The invoice reports this:

- The list of the loans ready to be purchased
- The loans retained by the bank
- Accrued interest
- The volume fee (if relevant)
- The sum total of funds to be paid to CRB

The purchaser sends the funds to CRB on loan origination day, or in the case of an ACH transfer, the day before.

The sum of the funds equals the total amount of the loans + the accrued interest + any bank fees (if relevant).

Note: The pre-purchase reports are available at 0600 Eastern on the SFTP location provisioned by CRB at the following location: \Prod\MPL\DailyReports\PreLoanSaleReports

Post-purchase report (receipt)

After the MPL or the investor funds CRB for the purchase, a post-purchase report, a receipt, is sent. The report confirms receipt of the funds and that the account has been reconciled.

The post-purchase reports contains this information:

- A list of the loans sold to them
- Fees and Interest paid to the bank.

Note: The post-purchase reports are made available immediately post sell on the SFTP location provisioned by CRB at the following location: \Prod\MPL\DailyReports\PostLoanSaleReports

Accrued interest is the interest that accumulated on the loan since the loan was funded. So, for example, a loan for $12,000 was made at an interest rate of 24.99%. CRB holds the loan for 3 days before it is sold.

This information is required to calculate the accrued interest:

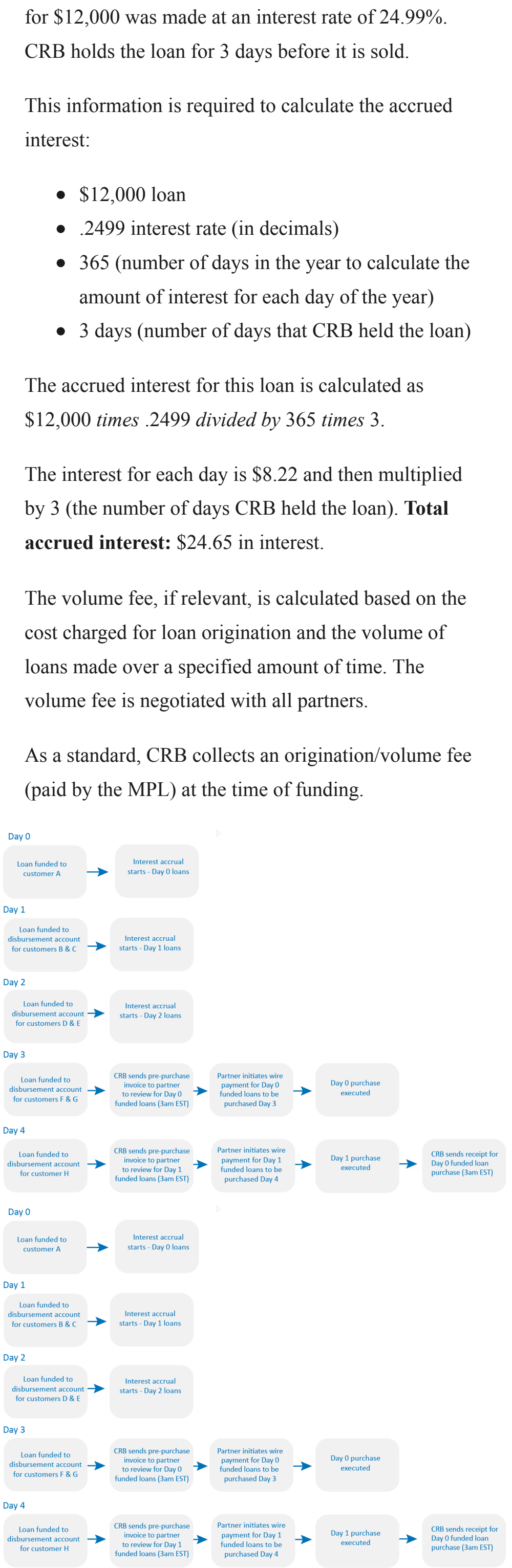
- $12,000 loan
- .2499 interest rate (in decimals)
- 365 (number of days in the year to calculate the amount of interest for each day of the year)
- 3 days (number of days that CRB held the loan)

The accrued interest for this loan is calculated as $12,000 *times* .2499 *divided by* 365 *times* 3.

The interest for each day is $8.22 and then multiplied by 3 (the number of days CRB held the loan). **Total accrued interest:** $24.65 in interest.

The volume fee, if relevant, is calculated based on the cost charged for loan origination and the volume of loans made over a specified amount of time. The volume fee is negotiated with all partners.

As a standard, CRB collects an origination/volume fee (paid by the MPL) at the time of funding.



Day 0
Loan funded to customer A
Interest accrual starts - Day 0 loans
Day 1
Loan funded to disbursement account for customers B & C
Interest accrual starts - Day 1 loans
Day 2
Loan funded to disbursement account for customers D & E
Interest accrual starts - Day 2 loans
Day 3
Loan funded to disbursement account for customers F & G
CRB sends pre-purchase invoice to partner to review for Day 0 funded loans (3am EST)
Partner initiates wire payment for Day 0 funded loans to be purchased Day 3
Day 0 purchase executed
Day 4
Loan funded to disbursement account for customer H
CRB sends pre-purchase invoice to partner to review for Day 1 funded loans (3am EST)
Partner initiates wire payment for Day 1 funded loans to be purchased Day 4
Day 1 purchase executed
CRB sends receipt for Day 0 funded loan purchase (3am EST)

- At funding - estimated volume fee is collected
- Purchase = Par + accrued Interest
- Day - Calendar day. Purchases take place on the 1st business day following a 3-day seasoning period.
- If ACH is used for purchases, settlement is T+1. Wires settle the same day.
- Purchase account – A CRB account to be provided, post board approval. The Routing # is 021214891
- For MPLs - File name format: LoanSales_PlatformName_yyyymmdd.xlsx
- For Investors - File name format: LoanSales_InvestorName_yyyymmdd.xlsx

Loan APIs

Arix includes a robust set of API calls to allow you to create, update, and manage your loans.

You can use cURL from the command line or a service such as Postman in our sandbox to test the API calls. We added images from Postman to demonstrate API requests and responses.

- The base URL for API calls to the Arix Sandbox environment is: https://arixapisandbox.crbnj.net
- The full API Swagger reference documentation is available at this URL: https://arixapisandbox.crbnj.net/swagger-ui.

Where to start

Go to Arix authentication to begin.

Request URL
https:// ... /api/Loan

API reference documentation

Click on each of the links for API reference documentation. These calls create loans and update loan information.

Create a loan **POST** /Loan

Add loan documents **POST** /Loan/{Id}/Attachments

Update loan funding info **PUT** /Loan/{Id}/FundingInfo

Update a loan by ID **PUT** /Loan/{Id}

Get loan details by ID **GET** /LoanDetail/{Id}

Get a summary of loans **GET** /LoanSummary/

Cancel a loan **PUT** /Loan/{{loanId}}/Cancel

Troubleshooting

The Swagger documentation site is protected by a firewall. Make sure your IP addresses are on the CRB allowlist so you can access the Arix sandbox.

Updated about 1 month ago

← Sales flow

Import Arix API test collection →

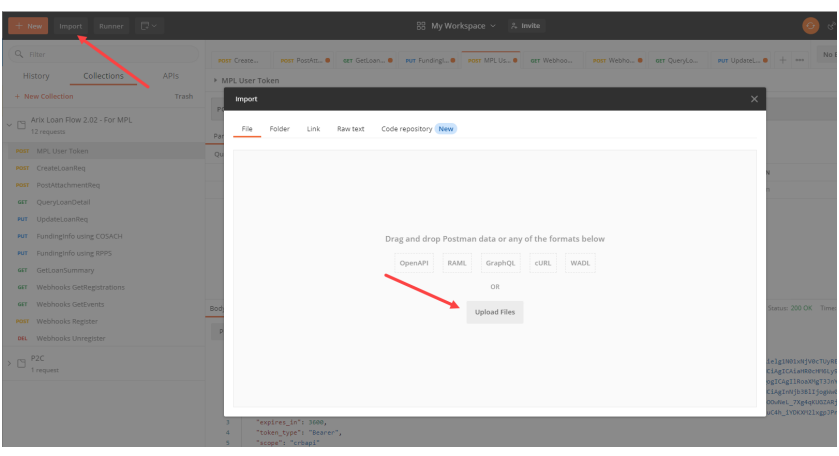
Did this page help you? Yes No

Import Arix API test collection

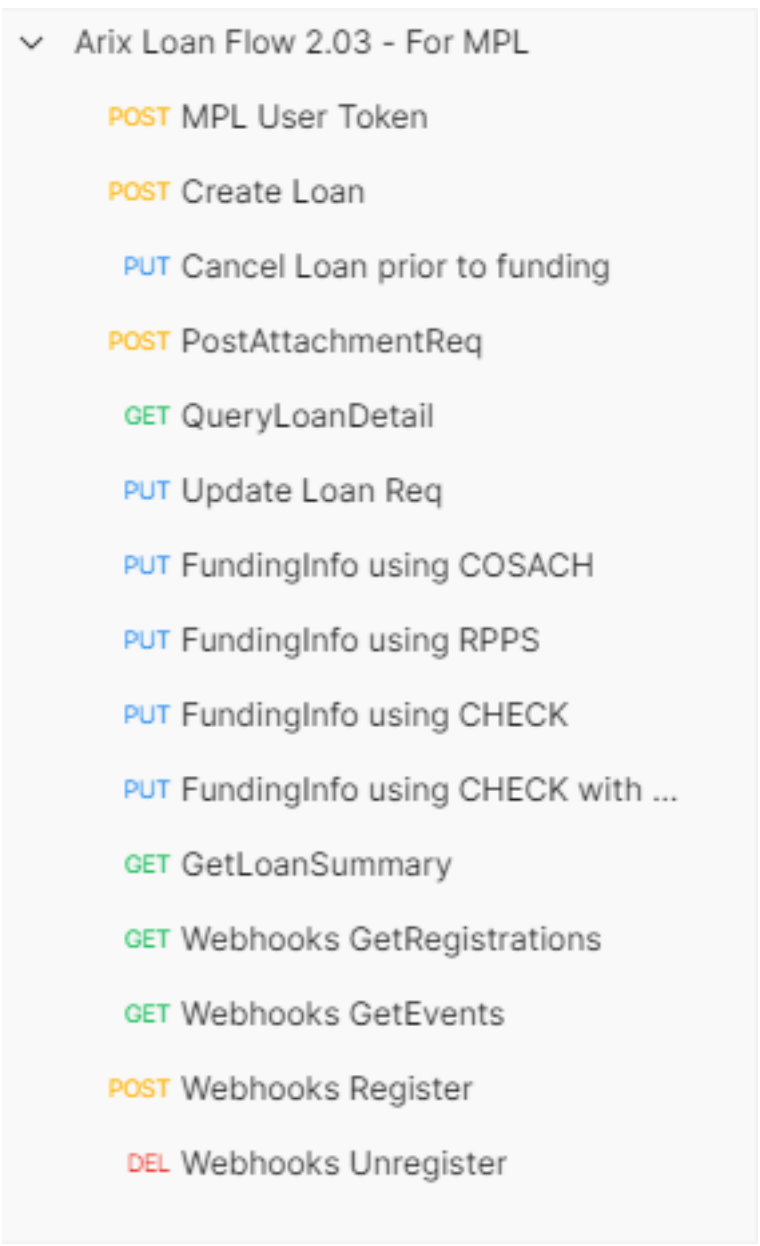
To start testing the Arix APIs in the sandbox environment, download the API test collection from the Monday board, or contact your Relationship Manager for access.

Below are images from Postman for demonstration.

1. Upload the test collection.
2. From the Postman UI, click **Import** -> **Upload Files**.
3. Navigate to the relevant file and click **Open**.



After you import it, the Arix test collection will show in the left menu.



Updated 18 days ago

← Loan APIs

CRB authentication →

Did this page help you? Yes No

CRB authentication

Cross River Bank (CRB) uses the oAuth open standard to enable authentication for access to API services. Before calling any of our exposed APIs, you need to receive a bearer token from the oAuth server. When you call APIs in the future, this is the token to copy and paste into your API calls. For more information, contact CRB.

> 📘
>
> **Important**
>
> The token expires after 60 minutes (the server reads 3600 seconds). If 60 minutes has passed and you want to call more APIs, you have to call the token again. You will receive the same token in return.

These are the addresses to call APIs for the CRB authentication sites in the sandbox and production environments.

- Sandbox URL: https://oauthtest.crbnj.net/connect/token
- Production URL: https://oauth.crbnj.net/connect/token

Sample access token call in cURL

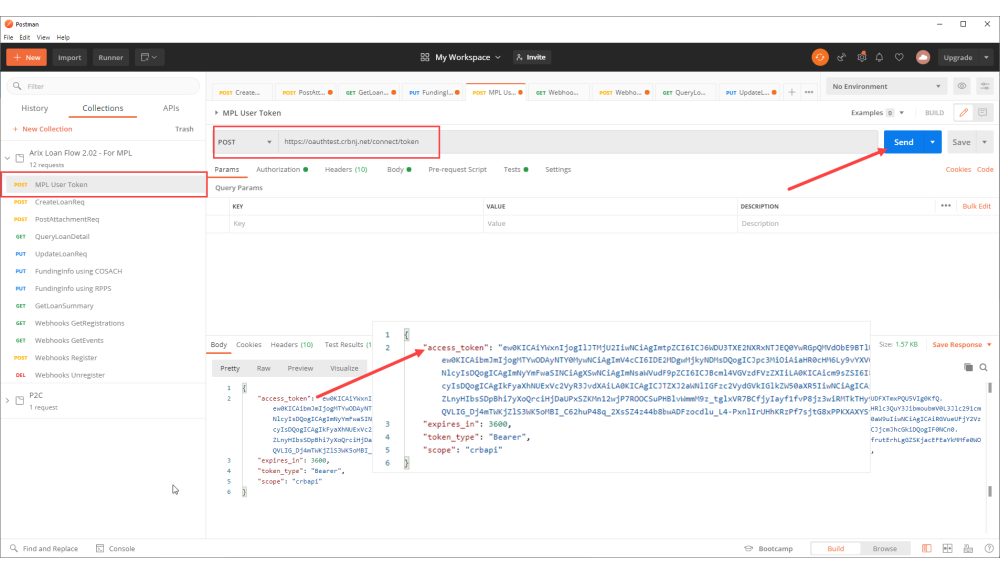
```
curl -X POST  https://oauthtest.crbnj.net/c
xxxxxxxxxx'  -H 'cache-control: no-cache'  -
urlencoded'  -d 'grant_type=client_credenti
```

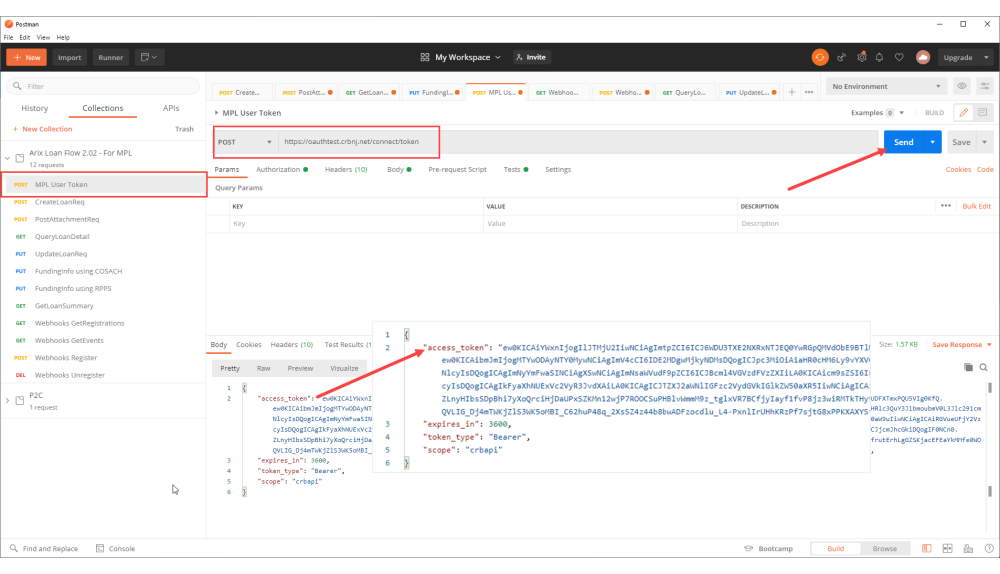
If the authentication is successful, the **Status Code** is **OK**.

Sample request for an access token in Postman

1. From the test collection you imported, click **POST MPL User Token**.
2. In the **Authorization** tab, enter the username and password provided to you by CRB.
3. Click **Send**.

If your credentials were entered correctly, an **access_token** shows in the body of the response, also referred to as a **bearer token**.

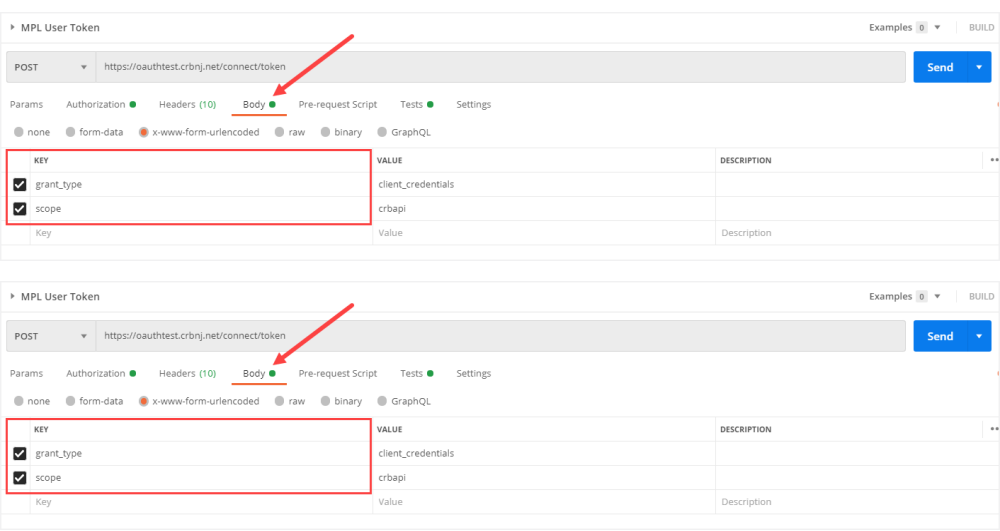




The x-www-form-urlencoded body

These are the criteria you should see. The x-www-form-urlencoded body must contain the following key-value pairs:

- **grant_type/ client_credentials**
- **scope/crbapi**



- If you can't get a bearer token and you **haven't** received one in the past, take these steps:
  - Confirm the URL.
  - Check that the username and password are typed correctly. **They are case sensitive**.
  - Check your username and password against the one you received from CRB.
  - If the credentials are correct, ensure that the password hasn't expired.
  - Contact CRB if your account is locked as a result of 3 incorrect log in attempts.
    If you are still having trouble, contact CRB to obtain valid credentials.
- If you can't get a bearer token and you **have** received one in the past, contact CRB.

Create a loan

Use this call to create a loan request in Arix.

Request URL
https://…/Loan

For detailed reference information on the loan attributes, see Loan attributes and parameters.

> **Important:**
>
> When you add a date field in a request, the format must be: yyyy-mm-dd.
>
> These fields include BorrowerDOB and FICODate and must be in this format: 1970-12-05

Sample request in JSON - POST /Loan

```
{
  "Loan": {
    "LoanNumber": {{loanNumber}},
    "IssuingBankId": "CRB",
    "Platform": "string",
    "LoanType": 1,
    "NoteDate": "2019-12-31",
    "BorrowerLastName": {{$randomLastName}}
    "BorrowerFirstName": {{$randomFirstName
    "BorrowerDOB": "1986-12-31",
    "BorrowerSSN": {{borrowerSSN}},
    "BorrowerAddress": {{borrowerAddress}},
    "BorrowerCity": {{$randomCity}},
    "BorrowerState": {{borrowerState}},
    "BorrowerZip": {{borrowerZip}},
    "LoanAmount": 1500,
    "NetFunding": 1300,
    "Rate": 0.2,
    "APR": "10",
    "BatchId": "string",
    "BorrowerPhone": {{$randomPhoneNumber}}
    "BorrowerEmail": {{$randomEmail}},
    "Amortization": 0,
    "Term": 36,
    "RateType": 0,
    "PriorLoanFlag": "string",
    "FICO": 10,
    "FICODate": "2017-12-31",
    "CreditGrade": "string",
    "AssetClass": 0,
    "LoanPurpose": "string",
    "Investor": "string",
    "Servicer": "string",
    "ApplicationDate": {{$applicationDate}},
    "RegBDecisionDate": {{RegBDecisionDate}
    "PurchaseDate": {{$purchaseDate}},
    "PaymentDueDate": {{PaymentDueDate}},
    "ApprovedLoanAmount": 0,
    "PeriodicPayment": 0,
    "OriginationFee": 100,
    "HomeownerFlag": "string",
    "OpenCreditLines": 0,
    "AnnualIncome": 0,
    "CreditInquiries12months": 0,
    "DQPast24months": 0,
    "PublicRecordsSnFile": 0,
    "EmploymentLength": 0,
    "DebtUtilization": 0,
    "TotalRevolvingDebt": 0,
    "AccountsOpenedPast24months": 0,
    "CollectionsExcludingMedical": 0,
    "MonthsSinceLastRecord": 0,
    "Employer": "string",
    "DTI": 0,
    "MLAFLAG": "string",
    "MAPR": "string",
    "ReceiverName": "string",
    "SameDayFlag": 0,
    "AddendaRecord": "string",
    "CoBorrowerFirstName": "string",
    "CoBorrowerLastName": "string",
    "CoBorrowerDOB": "1986-12-31",
    "CoBorrowerSSN": "string",
    "CoBorrowerAddress": "string",
    "CoBorrowerCity": "string",
    "CoBorrowerState": "string",
    "CoBorrowerZip": "string",
    "CoBorrowerPhone": "string",
    "CoBorrowerEmail": "string",
    "CoBorrowerFICO": 0,
    "MerchantName": "string",
    "MerchantFee": 0,
    "RefundAmount": 0,
    "AdjustedLoanAmount": 0,
    "AdjustedMNTFunding": 0,
    "AdjustedOriginationFee": 0,
    "AdjustedCurrentMonthlyPayment": 0,
    "RefundDate": "2017-12-31",
    "AdjustedMerchantFee": 0,
    "FinalPaymentDate": {{finalPaymentDate}}
    "FinalPayment": 0,
    "FinanceCharge": 0,
    "TotalPayments": 0,
    "PPY": 0,
    "PaymentFrequency": 0,
    "StandardEntryTypeCode": "string",
    "InterestAccrualMethodDays": 0,
    "SubpoolId": "Moe83",
    "Program": "DTM",
    "Field1": "string",
    "Field2": "string",
    "Field3": "string",
    "Field4": "string",
    "Field5": "string",
    "Field6": "string",
    "Field7": "string",
    "Field8": "string",
    "Field9": "string",
    "Field10": "string",
    "Field11": "string",
    "Field12": "string",
    "Field13": "string",
    "Field14": "string",
    "Field15": "string",
    "Field16": "string",
    "Field17": "string",
    "Field18": "string",
    "Field19": "string",
    "Field20": "string",
    "Rails": [
      {
        "ACHFields": {
          "StandardEntryClassCode": "PPD",
          "TransactionType": "push",
          "ToRoutingNumber": "021214891",
          "ToAccountType": "GL",
          "ToAccountName": "Test",
          "ToAccountNumber": "9999999999",
          "Description": "Test"
        },
        "RailType": "COSACH",
        "Priority": 1,
        "Amount": 300
      },
    ]
  }
}
```

Sample response in JSON - POST /Loan

```
{
    "Id": "bd054ec9-8598-497e-ac2d-ac5408bbb
    "MPLId": "TST-
    "Date": "/Date(1602674613963-0400)/",
    "LoanNumber": "M2-1afc220c66ee4ad",
    "IssuingBankId": "CRB",
    "Platform": "string",
    "LoanType": "HPS",
    "NoteDate": "/Date(1577768400000-0500)/"
    "BorrowerLastName": "Emard",
    "BorrowerFirstName": "Darrion",
    "BorrowerDOB": "/Date(536309720000-0500
    "BorrowerSSN": "640513072",
    "BorrowerAddress": "8652_VonRueden_Walk
    "BorrowerCity": "East",
    "BorrowerState": "LV",
    "BorrowerZip": "38078",
    "LoanAmount": 500,
    "NetFunding": 1300,
    "Rate": 0.05,
    "APR": 10,
    "BatchId": "string",
    "BorrowerPhone": "323-514-8396",
    "BorrowerEmail": "Helga_Hoeger14@gmail.
    "Amortization": 0,
    "Term": 36,
    "RateType": 0,
    "PriorLoanFlag": "string",
    "FICO": 10,
    "FICODate": "/Date(1514696400000-0500)/"
    "CreditGrade": "string",
    "AssetClass": 0,
    "LoanPurpose": "string",
    "Investor": "string",
    "Servicer": "string",
    "ApplicationDate": "/Date(1602674613934-
    "RegBDecisionDate": "/Date(1602674613934-
    "PurchaseDate": "/Date(1602087200000-040
    "PaymentDueDate": "/Date(1607922000000-0
    "ApprovedLoanAmount": 0,
    "PeriodicPayment": 0,
    "OriginationFee": 100,
    "HomeownerFlag": "string",
    "OpenCreditLines": 0,
    "AnnualIncome": 0,
    "CreditInquiries12months": 0,
    "DQPast24months": 0,
    "PublicRecordsSnFile": 0,
    "EmploymentLength": 0,
    "DebtUtilization": 0,
    "TotalRevolvingDebt": 0,
    "AccountsOpenedPast24months": 0,
    "CollectionsExcludingMedical": 0,
    "MonthsSinceLastRecord": 0,
    "Employer": "string",
    "DTI": 0,
    "MLAFLAG": "string",
    "MAPR": "string",
    "ReceiverName": "string",
    "SameDayFlag": 0,
    "AddendaRecord": "string",
    "CoBorrowerFirstName": "string",
    "CoBorrowerLastName": "string",
    "CoBorrowerDOB": "/Date(536309200000-05
    "CoBorrowerSSN": "string",
    "CoBorrowerAddress": "string",
    "CoBorrowerCity": "string",
    "CoBorrowerState": "string",
    "CoBorrowerZip": "string",
    "CoBorrowerPhone": "string",
    "CoBorrowerEmail": "string",
    "CoBorrowerFICO": 0,
    "MerchantName": "string",
    "MerchantFee": 0,
    "RefundAmount": 0,
    "AdjustedLoanAmount": 0,
    "AdjustedMNTFunding": 0,
    "AdjustedOriginationFee": 0,
    "AdjustedCurrentMonthlyPayment": 0,
    "RefundDate": "/Date(1514696400000-0500
    "AdjustedMerchantFee": 0,
    "FinalPaymentDate": "/Date(1618372800000
    "FinalPayment": 0,
    "FinanceCharge": 0,
    "TotalPayments": 0,
    "PPY": 0,
    "PaymentFrequency": 0,
    "StandardEntryTypeCode": "string",
    "InterestAccrualMethodDays": 0,
    "SubpoolId": "Moe83",
    "Program": "DTM",
    "Field1": "string",
    "Field2": "string",
    "Field3": "string",
    "Field4": "string",
    "Field5": "string",
    "Field6": "string",
    "Field7": "string",
    "Field8": "string",
    "Field9": "string",
    "Field10": "string",
    "Field11": "string",
    "Field12": "string",
    "Field13": "string",
    "Field14": "string",
    "Field15": "string",
    "Field16": "string",
    "Field17": "string",
    "Field18": "string",
    "Field19": "string",
    "Field20": "string",
    "Rails": [
        {
            "ACHFields": {
                "StandardEntryClassCode": "
                "TransactionType": "push",
                "ToRoutingNumber": "021214891
                "ToAccountType": "GL",
                "ToAccountName": "Test",
                "ToAccountNumber": "9999999
                "Description": "Test"
            },
            "Id": "c288ac03-803a-4b94-96c2-
            "RailType": "COSACH",
            "Priority": 1,
            "Amount": 300,
            "LoanId": "bd054ec9-8598-497e-a
            "MPLId": "TST-
            "IssuingBankId": "CRB",
            "isFailed": false,
            "LoanType": "Undefined"
        }
    ]
}
```

Add loan documents

Use this call to post loan documents.

Request URL
https:// ... /Loan/{Id}/Attachments

To support the loan approval process, send Arix a complete document package as one zip file. The package must include a signed loan agreement, a credit report, and other required documents as per your credit and underwriting policy, and your AML/BSA processes. To view the list of acceptable attachments, see Supporting loan documents.

> **Important**
>
> The status of a loan remains **Received** until all the required documents are received by Arix. No webhook is sent to report that docs are incomplete.

To add attachments:

1. Zip all the relevant documents.
2. Name the file with the **{Arix GUID}**.zip
3. **POST** the documents.

Click here for enum information.
Click here for information on the funding flow.

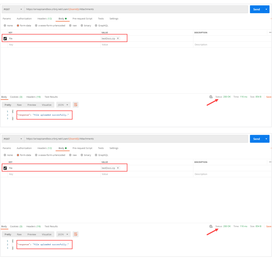
The naming convention requires that end of the file name (prior to the file extension) end with a dash (-) followed by a 2 letter code. Click here for the codes.

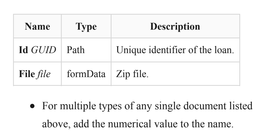
Example
- Loan agreement {*JoesCreditReport*}**-LA**.pdf
- Social Security card {*JoesCreditReport*}**-SC**.pdf

Notes for the zip files

- You can send multiple zip files
- Attachment size is set at 120 MB
- Files within the zip folder have to follow a naming convention (as shown above)

Sample request and response - POST /Loan/{Id}/Attachments





Name	Type	Description
Id *GUID*	Path	Unique identifier of the loan.
File *file*	formData	Zip file.

- For multiple types of any single document listed above, add the numerical value to the name.
  Example: For multiple paystubs (PS): 179317-**PS1** and 179317-**PS2**
- You can submit an additional 99 files as needed for underwriting, by following the convention, F1 - F99.
- If you want to submit an --F3 file, notify CRB ahead of time so we can identify F3 as a security video.
  Example: *MyRandomFile*-F3.mov

This table displays information on the documents received for the loan.

Attribute	Description
MPLId *string*	ID of the MPL.
LoanId *GUID*	Unique identifier for a loan.
FilePath	Path for retrieving the document. *Special permissions needed.*
FileName	Name of the file sent.
Date	Date the file was received.
FileOverwritten	If the original file was overwritten by another file, then the value will be 1.
IssuingBankId *string*	ID of the issuing bank.

Loan stuck in the status `Received`

A loan stays in `Received` for these reasons:

- The file wasn't named correctly based on regex verification
- Some or all required documents weren't received
- There was a problem unzipping the files. If a file is corrupt, Arix won't be able to unzip it. Fix it and resend a POST with the attachments.

Confirming document status

To confirm the document status for a loan, call GET LoanDetail and use these collections:

- **RequiredDocs** - to see which documents are required for a loan and if they were received.
- **Attachments** - to see which documents were received by Arix

You can add documents if a loan is in one of the statuses below. If a document is sent after the loan is funded, Arix will store it, but it won't parse it or run it through compliance.

Status	Description
1	Received
2	DocsComplete
3	PassedCompliance
4	Approved
11	AwaitingFunding
101	ComplianceFailed

Get a summary of loans

Use this call to retrieve a summary of loans in their current state.

Request URL
https:// ... /LoanSummary

For detailed reference information on the loan attributes, see Loan attributes and parameters.

Sample request in cURL - GET /LoanSummary

```
curl --location --request GET 'https://arix
--header 'Accept: application/json' \
--header 'Authorization: <BearerToken>' \
--header 'Cookie: ss-opt=temp; ss-id=w66BEN
```

Sample response in JSON - GET /LoanSummary

After you call a loan summary the **Results** that show are a list of the current loan statuses.

```
{
    "Offset": 0,
    "Total": 0,
    "Results": [
        {
            "Status": "Received",
            "TimeStamp": "/Date(16014682094
            "Id": "16ae6e4a-d96e-4151-8cbc-
            "MPLId": "TST",
            "Date": "/Date(1601468209427-00
            "LoanNumber": "AW-a68f17ec9244
            "IssuingBankId": "CRB",
            "Platform": "string",
            "LoanType": "HFS",
            "NoteDate": "/Date(1577768400000
            "BorrowerLastName": "Dach",
            "BorrowerFirstName": "Myles",
            "BorrowerDOB": "/Date(536389200
            "BorrowerSSN": "952953656",
            "BorrowerAddress": "963_Marco_F
            "BorrowerCity": "Labadieton",
            "BorrowerState": "SU",
            "BorrowerZip": "84720",
            "LoanAmount": 1500.000000,
            "NetFunding": 1300.000000,
            "Rate": 0.200000,
            "APR": 10.000000,
            "BatchId": "string",
            "BorrowerPhone": "933-392-5495"
            "BorrowerEmail": "Kaycee51@hotma
            "Amortization": 0,
            "Term": 36,
            "RateType": 0,
            "PriorLoanFlag": "string",
            "FICO": 10,
            "FICODate": "/Date(1514696400000
            "CreditGrade": "string",
            "AssetClass": 0,
            "LoanPurpose": "string",
            "Investor": "string",
            "Servicer": "string",
            "ApplicationDate": "/Date(16014
            "RegBDecisionDate": "/Date(1601
            "PurchaseDate": "/Date(16016976
            "PaymentDueDate": "/Date(160671
            "ApprovedLoanAmount": 0.000000,
            "PeriodicPayment": 0.000000,
            "OriginationFee": 100.000000,
            "HomeownerFlag": "string",
            "OpenCreditLines": 0,
            "AnnualIncome": 0,
            "CreditInquiries12months": 0,
            "DQPast24months": 0,
            "PublicRecordsOnFile": 0,
            "EmploymentLength": 0,
            "DebtToIlization": 0.000000,
            "TotalRevolvingDebt": 0.000000,
            "AccountsOpenedPast24months": 0
            "CollectionsExcludingMedical": 0
            "MonthsSinceLastRecord": 0,
            "Employer": "string",
            "DTI": 0.000000,
            "MLAFLAG": "string",
            "MAPR": "string",
            "ReceiverName": "string",
            "SameDayFlag": 0,
            "AddendaRecord": "string",
            "CoBorrowerFirstName": "string"
            "CoBorrowerLastName": "string",
            "CoBorrowerDOB": "/Date(5363892
            "CoBorrowerSSN": "string",
            "CoBorrowerAddress": "string",
            "CoBorrowerCity": "string",
            "CoBorrowerState": "string",
            "CoBorrowerZip": "string",
            "CoBorrowerPhone": "string",
            "CoBorrowerEmail": "string",
            "CoBorrowerFICO": 0,
            "MerchantName": "string",
            "MerchantFee": 0.000000,
            "RefundAmount": 0.000000,
            "AdjustedLoanAmount": 0.000000,
            "AdjustedNetFunding": 0.000000,
            "AdjustedOriginationFee": 0.000
            "AdjustedCurrentMonthlyPayment"
            "RefundDate": "/Date(1514696400
            "AdjustedMerchantFee": 0.000000
            "FinalPaymentDate": "/Date(1617
            "FinalPayment": 0.000000,
            "FinanceCharge": 0.000000,
            "TotalPayments": 0.000000,
            "PPY": 0,
            "PaymentFrequency": 0,
            "StandardEntryTypeCode": "string
            "InterestAccrualMethodDays": 0,
            "SubpoolId": "Moe83",
            "Program": "DTM",
            "Field1": "string",
            "Field2": "string",
            "Field3": "string",
            "Field4": "string",
            "Field5": "string",
            "Field6": "string",
            "Field7": "string",
            "Field8": "string",
            "Field9": "string",
            "Field10": "string",
            "Field11": "string",
            "Field12": "string",
            "Field13": "string",
            "Field14": "string",
            "Field15": "string",
            "Field16": "string",
            "Field17": "string",
            "Field18": "string",
            "Field19": "string",
            "Field20": "string",
            "Rails": [
                {
                    "ACHFields": {
                        "StandardEntryClass
                        "TransactionType": "
                        "ToRoutingNumber": "
                        "ToAccountType": "Ch
                        "ToAccountName": "Te
                        "ToAccountNumber": "
                        "Description": "Test
                    },
                    "Id": "da240b55-9a2b-40a
                    "RailType": "COSACH",
                    "Priority": 1,
                    "Amount": 300.0,
                    "LoanId": "16ae6e4a-d96e
                    "MPLId": "TST",
                    "IssuingBankId": "CRB",
                    "IsFailed": false,
                    "LoanType": "Undefined"
                }
            ]
        }
    ],
    "Meta": {}
}
```

- You can customize the result set with the `jsconfig` parameter.
- DateTime values are returned in the Unix format.

Use these parameters to design and filter a query to narrow down and present the most precise results possible.

Skip *integer or null*	**Enter a number of responses to skip.**
Take *integer or null*	Enter a number of responses to return.
OrderBy *string*	The field by which to sort.
OrderByDesc *string*	The field by which to sort descending.
Include *string*	Aggregate queries to include.
Meta *Dictionary*	A meta object.

Get a summary of loans

Use this call to retrieve a summary of loans in their current state.

Request URL
https:// ... /LoanSummary

For detailed reference information on the loan attributes, see Loan attributes and parameters.

Sample request in cURL - GET /LoanSummary

```
curl --location --request GET 'https://arix
--header 'Accept: application/json' \
--header 'Authorization: <BearerToken>' \
--header 'Cookie: ss-opt=temp; ss-id=w66BEN
```

Sample response in JSON - GET /LoanSummary

After you call a loan summary the **Results** that show are a list of the current loan statuses.

```
{
    "Offset": 0,
    "Total": 0,
    "Results": [
        {
            "Status": "Received",
            "TimeStamp": "/Date(16014682094
            "Id": "16ae6e4a-d96e-4151-8cbc-
            "MPLId": "TST",
            "Date": "/Date(1601468209427-00
            "LoanNumber": "AW-a68f17ec9244
            "IssuingBankId": "CRB",
            "Platform": "string",
            "LoanType": "HFS",
            "NoteDate": "/Date(1577768400000
            "BorrowerLastName": "Dach",
            "BorrowerFirstName": "Myles",
            "BorrowerDOB": "/Date(536389200
            "BorrowerSSN": "952953656",
            "BorrowerAddress": "963_Marco_F
            "BorrowerCity": "Labadieton",
            "BorrowerState": "SU",
            "BorrowerZip": "84720",
            "LoanAmount": 1500.000000,
            "NetFunding": 1300.000000,
            "Rate": 0.200000,
            "APR": 10.000000,
            "BatchId": "string",
            "BorrowerPhone": "933-392-5495"
            "BorrowerEmail": "Kaycee51@hotma
            "Amortization": 0,
            "Term": 36,
            "RateType": 0,
            "PriorLoanFlag": "string",
            "FICO": 10,
            "FICODate": "/Date(1514696400000
            "CreditGrade": "string",
            "AssetClass": 0,
            "LoanPurpose": "string",
            "Investor": "string",
            "Servicer": "string",
            "ApplicationDate": "/Date(16014
            "RegBDecisionDate": "/Date(1601
            "PurchaseDate": "/Date(16016976
            "PaymentDueDate": "/Date(160671
            "ApprovedLoanAmount": 0.000000,
            "PeriodicPayment": 0.000000,
            "OriginationFee": 100.000000,
            "HomeownerFlag": "string",
            "OpenCreditLines": 0,
            "AnnualIncome": 0,
            "CreditInquiries12months": 0,
            "DQPast24months": 0,
            "PublicRecordsOnFile": 0,
            "EmploymentLength": 0,
            "DebtToIlization": 0.000000,
            "TotalRevolvingDebt": 0.000000,
            "AccountsOpenedPast24months": 0
            "CollectionsExcludingMedical": 0
            "MonthsSinceLastRecord": 0,
            "Employer": "string",
            "DTI": 0.000000,
            "MLAFLAG": "string",
            "MAPR": "string",
            "ReceiverName": "string",
            "SameDayFlag": 0,
            "AddendaRecord": "string",
            "CoBorrowerFirstName": "string"
            "CoBorrowerLastName": "string",
            "CoBorrowerDOB": "/Date(5363892
            "CoBorrowerSSN": "string",
            "CoBorrowerAddress": "string",
            "CoBorrowerCity": "string",
            "CoBorrowerState": "string",
            "CoBorrowerZip": "string",
            "CoBorrowerPhone": "string",
            "CoBorrowerEmail": "string",
            "CoBorrowerFICO": 0,
            "MerchantName": "string",
            "MerchantFee": 0.000000,
            "RefundAmount": 0.000000,
            "AdjustedLoanAmount": 0.000000,
            "AdjustedNetFunding": 0.000000,
            "AdjustedOriginationFee": 0.000
            "AdjustedCurrentMonthlyPayment"
            "RefundDate": "/Date(1514696400
            "AdjustedMerchantFee": 0.000000
            "FinalPaymentDate": "/Date(1617
            "FinalPayment": 0.000000,
            "FinanceCharge": 0.000000,
            "TotalPayments": 0.000000,
            "PPY": 0,
            "PaymentFrequency": 0,
            "StandardEntryTypeCode": "string
            "InterestAccrualMethodDays": 0,
            "SubpoolId": "Moe83",
            "Program": "DTM",
            "Field1": "string",
            "Field2": "string",
            "Field3": "string",
            "Field4": "string",
            "Field5": "string",
            "Field6": "string",
            "Field7": "string",
            "Field8": "string",
            "Field9": "string",
            "Field10": "string",
            "Field11": "string",
            "Field12": "string",
            "Field13": "string",
            "Field14": "string",
            "Field15": "string",
            "Field16": "string",
            "Field17": "string",
            "Field18": "string",
            "Field19": "string",
            "Field20": "string",
            "Rails": [
                {
                    "ACHFields": {
                        "StandardEntryClass
                        "TransactionType": "
                        "ToRoutingNumber": "
                        "ToAccountType": "Ch
                        "ToAccountName": "Te
                        "ToAccountNumber": "
                        "Description": "Test
                    },
                    "Id": "da240b55-9a2b-40a
                    "RailType": "COSACH",
                    "Priority": 1,
                    "Amount": 300.0,
                    "LoanId": "16ae6e4a-d96e
                    "MPLId": "TST",
                    "IssuingBankId": "CRB",
                    "IsFailed": false,
                    "LoanType": "Undefined"
                }
            ]
        }
    ],
    "Meta": {}
}
```

- You can customize the result set with the `jsconfig` parameter.
- DateTime values are returned in the Unix format.

Use these parameters to design and filter a query to narrow down and present the most precise results possible.

Skip *integer or null*	**Enter a number of responses to skip.**
Take *integer or null*	Enter a number of responses to return.
OrderBy *string*	The field by which to sort.
OrderByDesc *string*	The field by which to sort descending.
Include *string*	Aggregate queries to include.
Meta *Dictionary*	A meta object.

Update a loan by ID

Use the loan ID in this call to update loan details. When you update field values in the LoanUpdate object of the request body, the response body will show the updated values.

Request URL
https:// ... /Loan/{Id}

Click here to access the call to update funding information.

To follow loan updates, query the loan with this call, GET LoanDetails.

>
>
> ### Information
>
> You can update loan details with **PUT** /Loan when the loan is in one of these statuses:

- Received (before loan passes compliance)
- ComplianceFailed (before loan passes compliance)
- AwaitingFunding (after the loan has passed compliance)

> If you send **PUT** /Loan in any other status, Arix won't update the loan, even if a 200 is returned.

For detailed reference information on the loan attributes in the LoanUpdate object, see Loan attributes and parameters.

Original Post /Loan API response


```
"Results": [
    {
        "Status": "Received",
        "TimeStamp": "/Date(1611047069948-0000)/",
        "MPLId": "TST",
        "Id": "298cae22-76d9-4a33-b0e9-acb500958d12",
        "Date": "/Date(1611047069923-0000)/",
        "LoanNumber": "LR-96e3f915eab244f",
        "IssuingBankId": "CRB",
        "Platform": "string",
        "LoanType": "HFS",
```

```
15          "NoteDate": "/Date(1577768400000-0000)/",
16          "BorrowerLastName": "Trantow",
17          "BorrowerFirstName": "Elenora",
18          "BorrowerDOB": "/Date(536389200000-0000)/",
19          "BorrowerSSN": "666712488",
20          "BorrowerAddress": "171_Jettie_Mountains",
21          "BorrowerCity": "Gislasonchester",
22          "BorrowerState": "TW",
23          "BorrowerZip": "78948",
```

```
4       "Results": [
5           {
6               "Status": "Received",
7               "TimeStamp": "/Date(1611047069948-0000)/",
8               "MPLId": "TST",
9               "Id": "298cae22-76d9-4a33-b0e9-acb500958d12",
10              "Date": "/Date(1611047069923-0000)/",
11              "LoanNumber": "LR-96e3f915eab244f",
12              "IssuingBankId": "CRB",
13              "Platform": "string",
14              "LoanType": "HFS",
15              "NoteDate": "/Date(1577768400000-0000)/",
16              "BorrowerLastName": "Trantow",
17              "BorrowerFirstName": "Elenora",
18              "BorrowerDOB": "/Date(536389200000-0000)/",
19              "BorrowerSSN": "666712488",
20              "BorrowerAddress": "171_Jettie_Mountains",
21              "BorrowerCity": "Gislasonchester",
22              "BorrowerState": "TW",
23              "BorrowerZip": "78948",
```


Updated API request using PUT /Loan/{Id}

The updated **BorrowerLastName** and **BorrowerFirstName** show below.

Use LoanUpdate object

When you update a loan with **PUT** /Loan, use a **LoanUpdate** object.

The attributes for **LoanUpdate** object are the same as those used in the **Loan** object, when you call **POST** /Loan.


```
{
  "LoanUpdate": {
    "LoanNumber": "LR-96e3f915eab244f",
    "IssuingBankId": "CRB",
    "Platform": "string",
    "LoanType": "HFS",
    "NoteDate": "/Date(1611047069923-0000)/'
    "BorrowerLastName": "Andersen",
    "BorrowerFirstName": "Betty",
    "BorrowerDOB": "/Date(536389200000-0000
    "BorrowerSSN": "666712488",
    "BorrowerAddress": "171_Jettie_Mountains
    "BorrowerCity": "Gislasonchester",
    "BorrowerState": "TX",
    "BorrowerZip": "78948",
    "LoanAmount": 1400,
```

```
"NetFunding": 800,
"Rate": 0.2,
"APR": "10",
"BatchId": "string",
"BorrowerPhone": {{$randomPhoneNumber}},
"BorrowerEmail": {{$randomEmail}},
"Amortization": 0,
"Term": 36,
"RateType": 0,
"PriorLoanFlag": "string",
"FICO": 10,
"FICODate": "2017-12-31",
"CreditGrade": "string",
"AssetClass": 0,
"LoanPurpose": "string",
"Investor": "string",
"Servicer": "string",
"ApplicationDate": {{ApplicationDate}},
"RegBDecisionDate": {{RegBDecisionDate}},
"PurchaseDate": {{PurchaseDate}},
"PaymentDueDate": {{PaymentDueDate}},
"ApprovedLoanAmount": 0,
"PeriodicPayment": 0,
"OriginationFee": 100,
"HomeownerFlag": "string",
"OpenCreditLines": 0,
"AnnualIncome": 0,
"CreditInquiries12months": 0,
"DQPast24months": 0,
```

```
"PublicRecordsOnFile": 0,
"EmploymentLength": 0,
"DebtUtilization": 0,
"TotalRevolvingDebt": 0,
"AccountsOpenedPast24months": 0,
"CollectionsExcludingMedical": 0,
"MonthsSinceLastRecord": 0,
"Employer": "string",
"DTI": 0,
"MLAFLAG": "string",
"MAPR": "string",
"ReceiverName": "string",
"SameDayFlag": 0,
"AddendaRecord": "string",
"CoBorrowerFirstName": "string",
"CoBorrowerLastName": "string",
"CoBorrowerDOB": "2017-12-31T09:18:46.3
"CoBorrowerSSN": "string",
"CoBorrowerAddress": "string",
"CoBorrowerCity": "string",
"CoBorrowerState": "string",
"CoBorrowerZip": "string",
"CoBorrowerPhone": "string",
"CoBorrowerEmail": "string",
"CoBorrowerFICO": 0,
"MerchantName": "string",
"MerchantFee": 0,
"RefundAmount": 0,
"AdjustedLoanAmount": 0,
```

```
"AdjustedNetFunding": 0,
"AdjustedOriginationFee": 0,
"AdjustedCurrentMonthlyPayment": 0,
"RefundDate": "2017-12-31T09:18:46.377Z"
"AdjustedMerchantFee": 0,
"FinalPaymentDate": {{FinalPaymentDate}}
"FinalPayment": 0,
"FinanceCharge": 0,
"TotalPayments": 0,
"PPY": 0,
"PaymentFrequency": 0,
"StandardEntryTypeCode": "string",
"InterestAccrualMethodDays": 0,
"SubpoolId": "Moe83",
"Program": "DTM",
"Field1": "string",
"Field2": "string",
"Field3": "string",
"Field4": "string",
"Field5": "string",
"Field6": "string",
"Field7": "string",
"Field8": "string",
"Field9": "string",
"Field10": "string",
"Field11": "string",
"Field12": "string",
"Field13": "string",
"Field14": "string",
```

```
    "Field15": "string",
    "Field16": "string",
    "Field17": "string",
    "Field18": "string",
    "Field19": "string",
    "Field20": "string",
    "Rails": [
      {
        "ACHFields": {
          "StandardEntryClassCode": "PPD",
          "TransactionType": "push",
          "ToRoutingNumber": "021214891",
          "ToAccountType": "GL",
          "ToAccountName": "Test",
          "ToAccountNumber": "9999999999",
          "Description": "Test"
        },
        "RailType": "COSACH",
        "Priority": 1,
        "Amount": 300
        }
    ]
  }
}
```

Update loan funding information

PUT /Loan/{Id}/FundingInfo

Use this call to update loan funding information for a particular loan by adding an ID into your path parameter.

Request URL
https:// .../Loan/{ld}/FundingInfo

You can use this call to:

- Attempt disbursements to the borrower in case of returns or failures
- Make multiple disbursements
- Add rails

Funds can be sent as one net sum or in multiple disbursements. Arix automatically calculates the sum of the loan and adjusts the amount down (if necessary) so that no funds that exceed the approved loan amount are sent.

For detailed reference information on the loan attributes, see Loan attributes and parameters.

> 📘
>
> **Important**
>
> When you send a PUT call to update loan information, already processed rails and rails that are in the process of being funded **aren't affected**.

Sample request in JSON - PUT /Loan/{Id}/FundingInfo

In this sample request, we used this API to update the loan amount from $5,000 to $7,000. We called the endpoint, `FundingInfoUpdate` to view the updated amount of $7,000.

```
{
  "FundingInfoUpdate": {
   "Rails": [
         {
    "RailType": "RPPS",
    "Priority": 1,
    "Amount": 7000,
    "RPPSFields": {
      "BillerID": "132654654632135423452",
      "BillerName": "Anita Loan",
      "ConsumerNameAtBiller": "TestNameAtBi
      "ConsumerAccountNumberAtBiller": "545
    }
  }
    ]
  }
}
```

Sample response in JSON - PUT /Loan/{Id}/FundingInfo

After calling **GET** /LoanDetail/{Id} to see if the loan was updated, the updated amount shows.

```
"Rails": [
                {
                    "RPPSFields": {
                        "BillerID": "132654(
                        "BillerName": "Anit
                        "ConsumerNameAtBill
                        "ConsumerAccountNuml
                    },
                    "Id": "a974ce4a-e7b3-43
                    "RailType": "RPPS",
                    "Priority": 1,
                    "Amount": 7000.0,
                    "LoanId": "ca485557-20c
                    "MPLId": "TST",
                    "IsFailed": false,
                    "LoanType": "Undefined"
                }
            ]
```

Cancel a loan

Use this call to cancel a loan by using its ID.

PUT /Loan/{{loanId}}/Cancel

If you want to cancel a loan the amount funded has to be zero, **amountfunded = 0**. If any of the rails have funded, it can no longer be canceled.

Once you cancel a loan, it goes into a terminal state of canceled and it can no longer be changed.

Request URL
https:// ... /Loan/{{loanId}}/Cancel

A body should not be submitted with this API call.

> **Important**
>
> To be able to cancel the loan, the loan must be in one of these statuses:
>
> - Received
> - DocsComplete
> - ComplianceFailed
> - Approved
> - AwaitingFunding
> - NotFullyFunded

Updated 19 days ago

← Update loan funding information

API reference material →

Did this page help you? Yes No

API reference material

The reference material for the API parameters are:

- Compliance results
- Supporting loan documents
- Document references
- Enums
- LoanDetail collections
- Loan funding information
- Loan attributes and parameters
- Response/Error codes
- Status updates
- Payment rails
  - ACH parameters
  - ACH returns
  - Adjustment Action parameters
  - Check parameters
  - RPPS parameters
  - RPPS returns
  - RPPS transaction reversals
  - XPay parameters

Updated 2 months ago

← Cancel a loan

API requests →

Did this page help you? Yes No

API reference material

The reference material for the API parameters are:

- Compliance results
- Supporting loan documents
- Document references
- Enums
- LoanDetail collections
- Loan funding information
- Loan attributes and parameters
- Response/Error codes
- Status updates
- Payment rails
  - ACH parameters
  - ACH returns
  - Adjustment Action parameters
  - Check parameters
  - RPPS parameters
  - RPPS returns
  - RPPS transaction reversals
  - XPay parameters

Updated 2 months ago

← Cancel a loan

API requests →

Did this page help you? Yes No

Compliance results

ComplianceLoanResult attributes

This section displays the compliance results of a loan.

Attribute	Description
ResultDate	Date of the result for the compliance check.
ComplianceRuleResultList	List of compliance rules, specific to each MPL.

ComplianceRuleResult parameters

These are the fields of a compliance result of a loan for each specific rule. If a rule failed, you can use these query parameters to check the *Data* against the *Rule* to see why it failed.

Parameter	Description
RuleName *string*	Name of the compliance rule. Specific for every MPL.
Rule *string*	Description of the compliance rule.
Data *string*	The actual data that was received for this loan in regards to this rule.
Result *string*	Result of the compliance check. True = Passed False = Failed

You can run a loan through compliance a second time with one of these calls:

- PUT /Loan{Id}/FundingInfo (to update funding information)
- PUT /Loan/{Id} (to update loan information)
- POST /Loan/{Id}/Attachments (to add attachments)

Updated about 2 months ago

← Response and error codes

Document reference →

Did this page help you? Yes No

Supporting loan documents

The documents below are acceptable as identification to support the loan process.

If you send in a loan with attachments and the loan remains in the status `Received`, instead of moving into the status `DocsComplete`, check the naming format.

The naming format of the attached documents is checked by a regex to verify that we received the required documents in the correct naming format. If we didn't receive the required documents or they weren't named properly, the loan will stay in `Received`.

The naming convention format of a document should be as follows: {ArivLoanGUID}-{Suffix}.{File Extension}

For example:
The credit report pdf for loan c5d3412a-f373-467b-b10f-7200e5e7582d should be named **c5d3412a-f373-467b-b10f-7200e5e7582d-CR.pdf**

For more information about adding loan documents, click here.

> **Note**
>
> We prefer that you send documents by XML or JSON. If necessary, you can send documents by PDF. The format needs to be agreed upon during the integration process.

Identification documents

Document type	Naming convention suffix
Credit report	-CR
Drivers license	-DL
Passport	-PP
Permanent resident card	-RC
Work Visa	-WorkVisa
Debit card *As secondary form of ID*	-DC
Credit card *As secondary form of ID*	-CC
Social Security card *As secondary form of ID*	-SC
Selfie *Used to verify the ID*	-SLF
State Issued ID	-SID
Proof of Address *jpeg, jpg, png or pdf provided as POA*	-POA
Rental Lease Agreement	-LeaseAgreement
Miscellaneous ID documents	-MID

Credit documents

Document type	Naming convention suffix
Credit report	-CR
Credit report (soft pull)	-CR-SP
Credit report (hard pull)	-CR_HP
Pagaya credit model	-CRM_PGY

Loan documents and disclosures

Document type	Naming convention suffix
Promissory note	-PN
Truth in Lending agreement	-TL
Loan Agreement	-LA (contains Truth in Lending agreement and Promissory Note)
Loan Application	-LApp
Preauthorized Transfer	-PreAuthTrn (for customer facing authorization requests)
Credit Score Disclosure Notice	-CreditScoreDisclosure
Miscellaneous loan documents	-MiscLoanDocs

Income verification documents

Document type	Naming convention suffix
Pay stub	-PS
Additional pay stub	-PayStubAdditional
Previous pay stub	-PayStubPrevious
Tax return	-TR
Tax line release	-TaxLienRelease
Form W2	-W2
Social Security Award letter (SSA)	-SS
Pension/retirement benefit statement	-RS
Bank statement	-BS
Bank transaction history	-BankTransactionHi
Credit card statement	-CreditCardStatem
Credit card balances	-CreditCardBalanc
Form 1099	-MI
Various income verification documents	-IV

Other documents for verification

Document type	Naming convention suffix
Voided check	-VC
Education-related document	-ED

Trusted 3rd party documents

Document type	Naming convention suffix
LexisNexis	-LN
LexisNexis FraudPoint	-LNF
LexisNexis InstantID	-LNI
LexisNexis OFAC	-LNO
Yodlee	-YOD
Equifax eID	-EID
TransUnion TLO	-TLO
Experian PreciseID	-PID
GIACT	-GCT
GIACT gVerify	-GV
GIACT gIdentity	-GI
Clarity	-CTY
Socure	-SO
Thomson Reuters Clear ID	-TRCID
Alloy Identity Decisioning Platform	-ALY
Onfido Facial Similarity	-OFS
SentiLink Synthetic - Fraud-Scores	-SNTLF
Cognito Identity Verification and Watchlist Screening	-CGNTIDWS
Bureau Van Dyke - Review	-BVDReview

Updated 28 days ago

Did this page help you? Yes No

Document reference

This section displays information on the documents that were received for the loan.

Attribute	Description
MPLId *string*	ID of the MPL.
LoanId *GUID*	Unique identifier of the loan.
FilePath	Path for retrieving the document. *Special permissions needed.*
FileName	Name of the file sent.
Date	Date the file was received.
FileOverwritten	If the original file was overwritten by another file, then the value will be 1.
IssuingBankId *string*	ID of the issuing bank.

RequiredDocumentStatus parameters

This section displays the received status of required documents.

Attribute	Description
DocumentType	One of the documents required for loan processing.
Received	Confirms if a particular document was received.
ReceivedDate	Date the document was received.

Confirming the document status

To confirm the document status for a loan, call GET LoanDetail and use these collections:

- **RequiredDocs** - to see which documents are required for a loan
- **Attachments** - to see which documents were received by Arix

Loan in the status `Received`

A loan stays in `Received` for these reasons:

- The naming format isn't correct based on a regex verification
- Required documents weren't received
- Required documents weren't named properly

> !
>
> **Important**
>
> The status of the loan remains **Received** until all the required documents are received by Arix. No webhook is sent to notify that docs are incomplete.

Troubleshoot:

1. Check if there is an error unzipping the files. If a file is corrupted, Arix won't be able to unzip it. Make sure to fix it and resend a POST with the attachments.
2. Check the log files for errors.

You can add documents if the loan is in one of the statuses below.

Status	Description
1	Received
2	DocsComplete
3	PassedCompliance
4	Approved
11	AwaitingFunding
101	ComplianceFailed

If a document is sent after the loan is funded, Arix will store it, but it will not parse it or run it through compliance.

Enums

LoanStatusEnum

These enums communicate the status of a loan.

Enum	Enum number	Description
`Received`	1	Arix received a new loan application. Send attachments as the next step in the process.
`DocsComplete`	2	All required documents were received.
`PassedCompliance`	3	Loan successfully passed compliance.
`Approved`	4	Loan was approved for funding.
`InFunding`	5	At least 1 rail was submitted for funding.
`Funded`	6	Full loan amount was funded.
`AccountingEntriesComplete`	7	*An internal CRB process.* Confirms that account entries are complete.
`ReadyToSell`	8	Loan finished the seasoning period and is ready to be sold.
`Sold`	9	Loan was sold.
`Settled`	10	Loan was paid back by end user (not in use).
`AwaitingFunding`	11	All compliance rules were passed and the system is waiting for the funding information. Once in this status, only a Put /fundingInfo will get the loan funded.
`InReverse`	80	A return/cancel has been requested, and the funding is being returned to the originating account.
`AccountingEntriesReversed`	90	*Internal status.* Loan was cancelled or fully returned and accounting entries were reversed.
`Cancelled`	100	Issuing bank cancelled the loan. **Terminal State**.
`ComplianceFailed`	101	Loan failed the compliance check.
`Rejected`	102	Issuing bank didn't approve - possible only prior to funding. **Terminal State**.
`Returned`	103	All rails that were successfully funded came back as returned.
`NotFullyFunded`	104	Funding was attempted, but a balance remains to be funded.

FundingResult enums

These enums communicate the results of a funding attempt.

Enum	Enum number	Description
`Success`	0	All funded.
`PartialFailure`	1	Some funds were sent successfully, but some failed.
`FundingFailed`	2	Funding failed and nothing was funded.

RailResultType enums

These enums communicate the results of a rail transaction as part of the funding attempt.

Enum	Enum number	Description
`NotAttempted`	0	Rail wasn't attempted.
`Requested`	1	API call was successful and the funds are pending transmission.
`Success`	2	Full requested amount has been funded and released on the rail. **Note:** There is lag time between the time an ACH is sent and when it is received (by the receiving bank), usually 1 business day (unless it was sent as a same-day ACH). CRB considers the rail a **success** if and until we receive a notice from the receiving bank that the rail failed. At that time, the status of the rail changes from **Success** to **Returned**.
`RequestFailure`	3	The request to fund failed immediately upon request (usually due to a mistake in the API instructions.)
`Rejected`	4	The rail was rejected.
`Returned`	5	The rail that was sent and marked as **success** was returned.
`Skipped`	6	A rail was skipped. No need to reattempt the funding.
`ValidationFail`	7	The rail failed because it couldn't be validated.
`CosAchPaymentRejected`	8	Arix received a notification that a specific ACH rail was rejected. *(Internal Ops)*
`CosAchPaymentCanceled`	9	CRB Ops cancelled the ACH payment. *(Internal Ops)*
`CosAchNocReceived`	10	A Notice of Change (NOC) was received. Relevant for recurrent disbursements. *(Internal Ops)* Contact customer for updated instructions.
`CosAchHoldEscalated`	11	CRB Ops put the ACH transaction on hold. *(Internal Ops)*
`CosCoreTransactionCompleted`	12	COS Core (part of XPay) transaction was completed successfully. *(Internal Ops)*
`CosCoreTransactionRejected`	13	COS Core (part of XPay) transaction was unable to complete. *(Internal Ops)*
`CosXPaymentRecieved`	15	Inbound payment request has been received and is awaiting approval. *(Internal Ops)*
`CosXPaymentCanceled`	16	Payment canceled at the sender's request or has expired. *(Internal Ops)*
`CosXPaymentRejected`	17	Payment was rejected by the receiver. *(Internal Ops)*
`CosXPaymentFailed`	18	Payment failed to post to either the sender or receiver's account. Most commonly due to insufficient funds. *(Internal Ops)*
`CosXPaymentCompleted`	19	Payment has been posted to both the sender and receiver's accounts. *(Internal Ops)*
`CosCoreTransactionCompletedReverse`	20	COS Core (part of XPay) reversal transaction has been completed. *(Internal Ops)*
`CosXPaymentRejectedReverse`	24	COS XPay requested reversal hasn't been approved. *(Internal Ops)*
`CosXPaymentFailedReverse`	25	COS XPay requested reversal was unable to be reversed. *(Internal Ops)*
`CosXPaymentCanceledReverse`	26	COS XPay reversal was canceled. *(Internal Ops)*
`CheckCashed`	27	Check was cashed.
`CheckPrinted`	28	Check was printed.
`DigitalCheckPrinted`	29	A digital check was printed.
`CheckMailed`	30	The check was mailed.
`Failed`	31	The check wasn't able to be cashed.

RailType enums

These enums communicate the type of rail used to make the transactions.

Enum	Enum number	Description
`RPPS`	4	RPPS rail.
`NF`	7	No funding.
`COSACH`	8	Next day funding via ACH.
`COSACHSD`	9	Same day funding via ACH.
`COSXPay`	10	Internal transfer between CRB accounts.
`AdjustmentAction`	20	Adjustments to a loan after it was funded
`Check`	30	Physical check.

RPPS enums

RPPS uses enums to return funding attempt results.

Enum name	Enum number	Description
`Received`	1	The funding attempt was received by RPPS.
`Submitted`	2	The funding attempt was submitted by RPPS.
`Pending`	3	The funding attempt is pending.
`Approved`	4	The funding attempt was approved by RPPS.
`Rejected`	5	The funding attempt was rejected by RPPS.
`Failed`	6	The funding attempt failed in RPPS.
`Returned`	7	The rail was returned by RPPS.

Did this page help you? Yes No

LoanDetail collections

A loan collection contains a specific set of results, in response to a query that you add to your call.

When you want to access information (any details) about a loan, use the GET /LoanDetail/{Id} API. You can add 1 or more of the loan collections available in Arix (which are shown in the table below), which customizes the API response.

The API response always returns the **OriginalLoanInfo** (original details that were entered when the loan was created), and can also return the information.
Therefore, if you made changes to a loan, it's easy to view and compare any of the changes you made to the loan.

To check the collections you've called, look at the **Fields** in the path parameter that show after the equal (=) sign.

In the response below, only results for **FundingAttempts** and **ProcessedRails** are returned, which in this case doesn't return any information.
GET /LoanDetail/{Id}?
Fields=FundingAttempts,ProcessedRails

Curerls updated

!

Important

When you send a PUT call to these endpoints, only unprocessed rails are overwritten.

- PUT /Loan/{Id}
- PUT /Loan/{Id}/FundingInfo

Already processed rails and rails that are in the process of being funded aren't affected.

Loan collections	Description
LoanInfoUpdates *array*	View all updates made to the loan. To view a full list of the parameters returned, see **LoanInfoUpdates** below.
FundingInfoUpdates *array*	View all funding updates made to the loan. To view a full list of the parameters returned, see **FundingInfoUpdates** below.
FailedLoanInfoUpdate *array*	View information about a loan that failed. To view a full list of the parameters returned, see **FailedLoanInfoUpdate** below.
FailedFundingInfoUpdate *array*	View information about the loan funding that failed. To view a full list of the parameters returned, see "FailedFundingInfoUpdate**"** below.
ProcessedRails *array*	View the rails that were processed for a loan. To view a full list of the parameters returned, go to Payment rails.
FundingAttempts *array*	View the number of times funding was attempted. To view a full list of the parameters returned, see **FundingAttempts** below.
ReturnedRails *array*	View the rails that were returned. To view a full list of the parameters returned, go to Payment rails.
StatusUpdates *array*	View the status of the loan. To view a full list of the parameters returned, go to Status updates.
Attachments *array*	View the attachments that were submitted, or those that are missing. To view a full list of the parameters returned, go to Documents.
RequiredDocs *array*	View the documents that are required for the loan. To view a full list of the parameters returned, go to Documents.
ComplianceLoanResult *array*	View the compliance status of the loan. To view a full list of the parameters returned, go to Compliance results .
AdjustmentActions *array*	View updates made to the loan after funding. To view a full list of the parameters returned, go to Adjustment Action parameters.

LoanInfoUpdate collection parameters

This collection returns information about any updates made to a loan.

Attribute	Description
LoanInfoUpdateId *string*	Unique identifier for the loan update.
UpdatedBy *string*	Clerk who updated the loan.
UpdatedOn *DateTime*	The date the loan was updated

FundingInfoUpdate collection parameters

This collection returns information about any funding updates made to a loan.

Attribute	
LoanId *string*	Unique identifier for the loan.
MPLId *string*	ID of the MPL.
UpdatedBy *string*	Clerk who updated the loan.
UpdatedOn *DateTime*	The date the loan was updated.
Rails *List* [RailTransaction]	Click Payment rails for a list of rail types.

FundingAttempt collection parameters

This collection returns information about the attempts made to fund the loan, and the result.

Attribute	Description
LoanId *string*	Unique identifier of the loan.
Time *Time*	Time the funding attempt was made.
TransactionGroupId *string*	Group ID for the transaction.
FundingResult *FundingResult*	List of possible funding results. - Success - PartialFailure - FundingFailed See Enums for more information.
AmountFunded *decimal*	Dollar amount funded.

FailedLoanInfoUpdate collection parameters

This collection returns information about a loan that failed.

Attribute	Description
FailedLoanInfoUpdateId *string*	Unique identifier of information about the failed loan.
UpdatedBy *string*	Clerk who updated the loan.
UpdatedOn *DateTime or null*	The date the loan was updated.
ErrorMessages *string*	Error message.

FailedFundingInfoUpdate collection parameters

This collection returns information about loan funding that failed.

Attribute	Description
ErrorMessages *string*	Error message.

RequiredDocs collection parameters

This collection returns information about Required Documents and their received status for this loan.

Attribute	Description
DocumentType	One of the documents required for loan processing
Received	Confirms if a particular document was received.
ReceivedDate	Date the document was received.

Loan funding information

Funding information

This section displays the results of loan funding.

Attribute	Description
Id *string*	Unique identifier for the loan funding information.
LoanId *string*	Unique identifier for the loan.
Time *string*	Time that the information was returned.
TransactionGroupId *string*	ID for the transaction group.
FundingResult *string*	Indicates if the funding was successful.
AmountFunded *number*	Indicates the amount funded.

Failed loan

This section displays information for a failed loan.

Attribute	Description
FailedLoanInfoUpdateId *string*	The ID number of the failed loan info update.
UpdatedBy *string*	By whom the information was updated.
UpdatedOn *string*	Date/time the information was updated.
ErrorMessages *string*	Error message describing the reason for failure.

Updated about 1 year ago

← Loan attributes and parameters

Multiple disbursements →

Did this page help you? Yes No

Loan attributes and parameters

The table below is the reference information for the API attributes you can use when you create a loan or update a loan. They are the same attributes that show when you receive a response. Required response attributes show in the second table at the bottom.

Request Object: Loan (for POST loan)

Request Object: LoanUpdate (for PUT request)

Important:

When you add a date field in a request, the format must be: **yyyy-mm-dd**.

These fields include BorrowerDOB and FICODate and must be in this format: 1970-12-05

For fields that are of type **Number** (decimal), Arix stores 6 decimal points.
For fields that are of type **Integer**, you shouldn't send a decimal.

Attribute	Description
LoanNumber *string*	A customer generated loan number. Up to 100 characters. **Required**
IssuingBankId *string*	The ID of the issuing bank. CRB or BRB. **Required**
Platform *string*	The entity that is the intermediary between the customer and CRB. **Required**
LoanType *string*	Select from: 1 - HFS (Hold For Sale) 2 - LTHFS (Long Term Hold For Sale) 4 - RET (Retained) **Required**

NoteDate *string*	Date the borrower becomes obligated to pay the loan. Date can be the signing date or the funding date (dependent on the loan documents). The format must be: **yyyy-mm-dd**. **Required**
BorrowerLastName *string*	Borrower's last name. **Required**
BorrowerFirstName *string*	Borrower's first name. **Required**
BorrowerDOB *string*	Borrower's date of birth. The format must be: **yyyy-mm-dd**. **Required**
BorrowerSSN *string*	Borrower's Social Security Number. **Required**
BorrowerAddress *string*	Borrower's address. **Required**
BorrowerCity *string*	Borrower's city of residence. **Required**
BorrowerState *string (2 letter code)*	Borrower's State of residence. **Required**
BorrowerZip *string (5 digits)*	Borrower's zip code. **Required**
LoanAmount *number*	Gross dollar amount requested. This amount represents the sum of the origination fee plus (+) net funding amount. **Required**
NetFunding *number*	Net dollar amount to be funded. **Required**
Rate *number*	Interest rate, as a percent. **Required**

APR *number*	Annual percentage interest rate, as a percent. **Required**
ServicingBankId *string*	ID of the servicing bank. *Optional*
BorrowerPhone *string (10 digits)*	Borrower's phone number. *Optional*
BorrowerEmail *string*	Borrower's email address. *Optional*
Amortization *integer or null*	Number of payments at loan origination to pay off the loan. *Optional*
Term *integer or null*	Number of months from loan origination to pay off the loan. *Optional*
RateType *integer or null*	1 - fixed 2- floating *Optional*
PriorLoanFlag *string*	Red-flag issues in the past, such as bounced checks or insufficient funds. Y - yes N - no *Optional*
FICO *integer or null*	Borrower's 3-digit FICO score used for risk assessment. *Optional*
FICODate *string*	Date FICO score was obtained. The format must be: **yyyy-mm-dd**. *Optional*
CreditGrade *string*	Internal grade for loan or from credit policy. *Optional*

AssetClass *integer or null*	Reason for loan. 1 - student 2 - solar 3 - debt consolidation 4 - home improvement 5 - POS (point of service) 6 - medical 7 - auto 99 - other *Optional*
LoanPurpose *string*	Borrower's stated purpose for the loan. *Optional*
Investor *string*	Name of investor who will be purchasing the loan. *Leave blank for loans retained by CRB.* *Optional*
Servicer *string*	The entity that handles the servicing of your account, such as billing statements, notices. *Optional*
ApplicationDate *string*	Date the borrower applied for the loan. The format must be: **yyyy-mm-dd**. *Optional*
RegBDecisionDate *string*	Date the loan was approved. The format must be: **yyyy-mm-dd**. *Optional*
PurchaseDate *string*	Date the lender or investor will purchase the loan from CRB. The format must be: **yyyy-mm-dd**. *Optional*

PaymentDueDate *string*	Date the first payment is due. The format must be: **yyyy-mm-dd**. *Optional*
ApprovedLoanAmount *number*	Dollar amount the borrower was approved to borrow at the time of application. The amount can change before the final loan amount for reasons related to the borrower or lender. *Optional*
PeriodicPayment *number*	Dollar amount of monthly payments. *Optional*
OriginationFee *number*	Charge to the borrower for the loan. *Optional*
HomeownerFlag *string*	Confirm if the borrower is a homeowner. Yes or No. *Optional*
OpenCreditLines *integer or null*	Number of open credit lines. *Optional*
AnnualIncome *integer or null*	Borrower's annual income. *Optional*
CreditInquiries12months *integer or null*	Number of credit inquiries made on the borrower in the last 12 months. *Optional*
DQPast24months *integer or null*	Number of times the borrower has paid their loan late in the last 24 months on all accounts. *Optional*
PublicRecordsOnFile *integer or null*	Number of open public records, such as bankruptcy, tax liens or judgements. *Optional*
EmploymentLength *integer or null*	Number of years the borrower has been employed at their current job. *Optional*
DebtUtilization *number*	Percent of available credit used by borrower. *Optional*

TotalRevolvingDebt *number*	Dollar amount of all revolving debt on the credit report. *Optional*
AccountsOpenedPast24months *integer or null*	Number of accounts the borrower opened in the last 24 months. *Optional*
CollectionsExcludingMedical *integer or null*	Number of open collection accounts, not including collections for medical procedures. *Optional*
MonthsSinceLastRecord *integer or null*	This field indicates the number of months since the last active public record was on their credit report. A public record may indicate that a borrower stopped paying bills. *Optional*
Employer *string*	Name of employer. *Optional*
DTI *decimal*	**Debt to Income Ratio** Ratio of total monthly payment obligations divided by the monthly income as a decimal, not as a percent. It is calculated differently for each credit policy. It must be submitted as a **decimal**. *Optional*
MLAFLAG *string*	If the loan is being made to a covered borrower. Yes or No. *Optional*
MAPR *string*	**Military APR** A standard calculation used by lenders for loans covered by the Department of Defense's Military Lending Act & Regulation Z. MAPR is required if MLAFLAG = Yes For more information, see: https://www.fdic.gov/regulations/compliance/manual/5/V-13.1.pdf *Optional*

ReceiverName *string*	Name of party receiving funds via ACH. *Optional*
SameDayFlag *integer or null*	If the loan being funded via ACH or same-day ACH. (Same-day ACH requires set up by CRB). *Optional*
AddendaRecord *string*	Additional record used for information purposes. *Optional*
CoBorrowerFirstName *string*	Coborrower's first name. *Optional*
CoBorrowerLastName *string*	Coborrower's last name. *Optional*
CoBorrowerDOB *string*	Coborrower's date of birth. The format must be: **yyyy-mm-dd**. *Optional*
CoBorrowerSSN *string*	Coborrower's Social Security Number. *Optional*
CoBorrowerAddress *string*	Coborrower's address. *Optional*
CoBorrowerCity *string*	Coborrower's city of residence. *Optional*
CoBorrowerState *string (2 letter code)*	Coborrower's State of residence. *Optional*
CoBorrowerZip *string (5 digits)*	Coborrower's zip code. *Optional*
CoBorrowerPhone *string (10 digits)*	Coborrower's phone number. *Optional*
CoBorrowerEmail *string*	Coborrower's email address. *Optional*
CoBorrowerFICO *integer or null*	Coborrower's FICO score. *Optional*

MerchantName *string*	Merchant's name. *Optional*
MerchantFee *number*	Dollar amount paid to merchant. *Optional*
RefundAmount *number*	Refund amount. *Optional*
RefundDate *string*	Refund date. The format must be: **yyyy-mm-dd**. *Optional*
FinalPaymentDate *string*	Date due for final loan payment. The format must be: **yyyy-mm-dd**. *Optional*
FinalPayment *number*	Dollar amount of final payment. *Optional*
FinanceCharge *number*	Finance charge paid by the borrower. *Optional*
TotalPayments *number*	Total paid by the borrower after all the payments were made. *Optional*
PPY *number*	Number of payments per year. *Optional*
PaymentFrequency *Integer or null*	Frequency of payments: 1 - weekly 2 - semi-monthly 3 - monthly 4 - bi-monthly 5 - other 6 - bi-weekly *Optional*

StandardEntryTypeCode *string*	PPD - consumer accounts. CCD - corporate accounts. *Optional*
InterestAccrualMethodDays *integer or null*	365 (days). *Optional*
SubpoolId *string*	ID of the subpool that the loan belongs to. *Optional*
Program *string*	MPL defined program that the loan belongs to. *Optional*
Field1 *string*	Reserved for submitter's IP address *Required*
Field2 *string*	Reserved for "{ ""OFAC decision"":""True|False"", ""OFAC decision source"":"""" }" *Required*

Field3
string
	Reserved for
	"{
	""CIP Decision"":""NonDocumentary, Documentary,Other"",
	""CIP Vendors used"":{
	""CIP KYC vendor"":""Socure"",
	""CIP bank account vendor"":""Plaid""
	},
	""CIP Documents used"":{
	""CIP Document"":""Driving license"",
	""CIP Seconday Document"":""Utility Bill""
	},
	""CIP Verification used"":{
	""OCR"":""Mitek"",
	""Fraud"":""Experian Fraud"",
	""Phone number"":""White pages""
	}
	}"

	Required
Field4 *string*	Reserved for loan agreement version *Required*
Field5 *string*	Reserved for credit decisioning parameters and "{ "RevolvingLineFirstAppearance":"yes/no", "RevolvingLineGUID":"ID From Origination/First Appearance", "RevolvingLineBalanceAsOfWhenCallingCRB":" }" *Required*
Field6 *string*	**Required for business information** ***See standard json format below***
Field7 *string*	**Reserved for business owner Information** Collect information on controlling person and business owners who own 25% or more. *For SBA loans the threshold is 20%.* ***See standard json format below***
Field8 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field9 *string*	**Reserved for fraud vendor raw response** Up to 8,000 characters. *Optional*
Field10 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field11 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field12 *string*	Open field for text. Up to 8,000 characters. *Optional*

Field13 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field14 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field15 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field16 *string*	**Reserved for second look investors raw response** Up to 8,000 characters. *Optional*
Field17 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field18 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field19 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field20 *string*	Open field for text. Up to 8,000 characters. *Optional*
Field21 *byte [ ]*	Optional addition of loan information. Byte array in Base 64. Up to 8,000 characters. *Optional*
Rails *List {RailTransaction}*	Click rail transactions for information on rails. *Optional*


```
{
    "name": "business name",
    "tin": "EIN/SSN",
    "incDate": "2008-01-01",
    "stateIncorp": "AZ",
```

```
    "countryDomicile": "US",
    "phone": "4805477526",
    "email": "business email",
    "address": "business address",
    "city": "business city",
    "state": "business state",
    "zipCode": "business zip",
    "payroll": "Average monthly payroll",
    "businessType": "Business Entity Type - Sole proprietor,Partnership, C-Corp,
    "businessPurpose": "Business purpose / description",
    "industry": "xxx",
    "naicsCode": "xxx",
    "franchise": "Yes/No",
    "RegisteredTradingNames": "",
    "ProductServicesOffered": "",
    "InsuranceType": "",
    "AuthorisedSignatory": "",
    "AuthorisedSignatoryVerificationMethod": "",
    "BusinessLicenseInformation": "",
    "BusinessWebSiteVerification": "Yes|No",
    "BusinessAnnualTurnOver": "",
    "BusinessPricingDetails": {
        "pricingTier1": "",
        "pricingTier2": "",
    "pricingTier3": ""
    },
    "BusinessCreditLimit": {
        "ConsumerCreditLimit": "",
        "DailyCreditLimit": "",
        "BusinessApprovalException": "Y|N"
    }
}



[
    {
        "name": "owner Name",
        "ownershipPer": 24,
        "tin": "SSN|EIN|TIN",
        "address": "owner address",
        "city": "owner city",
        "state": "ownerstate",
        "zipCode": "owner zip",
        "dob": "YYYY-MM-dd",
        "phone": "owner phone",
                "OwnerFico": 700,
        "CIP Decision": "NonDocumentary, Documentary,Other",
        "CIP Vendors used": {
            "CIP KYC vendor": "eg Socure",
            "CIP bank account vendor": "eg Plaid"
        },
```

```
        "CIP Documents used": {
            "CIP Document": "eg for a person a Driving license",
            "CIP Secondary Document": "eg for a person Utility Bill"
        },
        "CIP Verification used": {
            "OCR": "eg Mitek",
            "Fraud": "eg Experian Fraud",
            "Phone number": "eg White pages",
            "type": "Owner | Control person",
            "citizenship": "US",
            "residencyType": "Permanent Resident Alien | Non-Immigrant Alien | I
        }
    },
    {}
]
```

Any vendors noted above, are provided for illustration.

Attribute	Description
Status *string*	The status of a loan. See Enums for response statuses.
TimeStamp *string*	Timestamp of the update.
Id *GUID*	Unique identifier for the loan.
MPLId *string*	ID of the MPL.
Date *string*	Date of the transaction.

Response and error codes

HTTP response codes

Arix uses standard HTTP response codes to communicate the result of a request.

HTTP Code	Description
200 series	Success. We successfully received your loan request. Arix will now validate the compliance rules set up for your program.
400	There is something wrong with the data in your request, or the request cannot be processed.
401	There is an issue with your API access token.
403	Your token is good, but you don't have permission to do what is being requested.
404	The resource specified doesn't exist.
500	Technical difficulties on our end. For more information, see Potential Issues (500 Series Error).

Errors

If something goes wrong with a call, we return a clear error object in the response.

Sample error response

```
{
  "Errors":[
    {
      "Code":1000,
      "Message":"First name required"
    },
    {
      "Code":1000,
      "Message":"Last name required"
    },
  ]
}
```

Error Codes

This is a list of the error code ranges.

Error Code Range	Description
1000 - 1999	Validation issue such as a missing required field.
2000 - 2999	Rule violation such as trying to add funds to a closed account.
3000 - 3999	Security issue with token or permissions.
9999	System issue such as an error on our end.

Updated 10 months ago

← API requests

Compliance results →

Did this page help you? Yes No

Status updates

This section returns information on the status of a loan. A loan flow will have multiple status updates.

Status Update parameters

Attribute	Description
LoanId *string*	Unique identifier for a loan.
Status *string*	Loan status. For more information, see Enums.
TimeStamp *string*	Timestamp for the update.
DateInserted *string*	Timestamp when the status was entered, (internal).

Updated 10 months ago

← Multiple disbursements

Supporting loan documents →

Did this page help you? Yes No

Payment rails

A payment rail is the method by which Arix transfers funds. Arix uses these rails:

- ACH
- Physical checks
- RPPS
- XPay

> 📘
>
> **Important**
>
> To track rails, use the **RailId** you receive back after the API call.

These properties are returned for rail transactions.

Every rail includes a **GUID**, which is a unique identifier for the rail which you can use to match the rail request to the rail response.

Attribute	Description
RailId **GUID**	Unique identifier for a rail.
LoanId *string*	Unique identifier for a loan.
RailType *RailType or null*	Type of rail for payment. Select from one of these rail types. - ACH *(COSACH and COSACHSD)* - Adjustment actions - Physical check - RPPS - XPay
Priority *integer or null*	Order the loan were processed - starts with number 1. If all the loans have the same number, Arix processes them in the order they were received.
Amount *decimal*	Amount attempted to process on this rail. If left blank, Arix attempts to process the full balance.
MPLId *string*	ID of the MPL.
IssuingBankId *string*	ID of the issuing bank.
IsFailed *boolean*	Indicates if the loan transmission failed. True or false.
LoanType *string*	Type of loan.
AmountFunded *decimal*	Dollar amount funded.
RailTransactionId *string*	Unique identifier for a rail transaction.
FundResult *RailResultType*	Result of the funding. See Enums.
Message *string*	Message returned.
ProcessedAt *DateTime*	Date the rail was processed.
TransactionGroupId *GUID*	Unique identifier of the transaction group.

The bank reserves 1.5 - 2 hours for pre-processing of funding before the network cutoff times below. Only loans that completed all underwriting checks are funded. Loans that miss a cutoff will auto-roll to the next processing window.

Rail type	Cutoff times
ACH	Cut off is 16:00 EST. The Fed accepts submissions until 18:00 EST.
ACH SD (Same Day) Limited to $100,000 per transaction (going up to $1,000,000 effective Mar/18/2022).	Cut off is 13:00 EST. The Fed accepts submissions until 14:45 EST. Extended Same Day cut off is 14:30 EST. The Fed accepts submissions until 16:45 EST.
Checks	Cut off is 15:00 EST.
Mastercard RPPS	Support for 3 MC RPPS processing cycles: 03:00 EST – 05:00 EST 13:00 EST – 15:00 EST 21:00 EST – 23:00 EST Cut off is 20 minutes before a cycle start time.
XPay	24/7 (no cutoff times)

ACH returns are received in batches, with the following cutoff times:

ACH returns	**1. First Batch: 12:00 AM EST** **2. Second Batch: 3:00 AM EST** **3. Third Batch: 11:30 AM EST** **4. Fourth Batch: 3:30 PM EST** **5. Fifth Batch: 4:30 PM EST**

ACH parameters

Arix uses ACH to push money into the account of a borrower, vendor, or creditor. CRB must approve every ACH payment that is sent.

There are two types of ACH rails:

- **COSACH**
  - Via NACHA
  - Funded next day
- **COSACHSD**
  - Via NACHA
  - Same day funding

JSON


```
{
    "LoanId": "bbedd717-8589-4411-b6da-
    "Rails": [
        {
            "ACHFields": {
                "StandardEntryClassCode
                "TransactionType": "pus
                "ToRoutingNumber": "021
                "ToAccountType": "Check
                "ToAccountName": "Anita
                "ToAccountNumber": "999
                "Description": "Loan"
            },
            "Id": "23b5fedb-bc20-41a0-a
            "RailType": "COSACH",
            "Priority": 1,
            "Amount": 100
        }
    ]
}
```


Request parameters

Attributes	Description
ToRoutingNumber *string*	Receiving bank's routing number. The value entered for this field is validated up front and it must be 9 digits long. **Required**
ToAccountName *string*	Name on the customer account. Limited to 2 characters. **Required**
ToAccountNumber *string*	Account number. Limited to 1 digits. **Required**
Description *string*	Mapped to identificatio field in the NACHA file Limited to 1 characters. **Required**
StandardEntryClassCode *string*	The type of account: - PPD - consumer accounts - CCD - corporate accounts **Required**
TransactionType *string*	Push. **Required**
ToAccountType *string*	Checking account or savings account. **Required**

Updated about 2 months ago

← Payment rails

ACH returns →

Did this page help you? Yes No

Check parameters

To pay down loan proceeds, you can send a physical check to a biller (merchant or business). This is useful when you have to pay billers that don't accept payments via Mastercard RPPS.

You can add an optional Base64 encoded attachment to accompany the check, and it will be mailed as a PDF.

To format the field **Remittance_advice** so that it reads correctly, follow the details points below:

- Quotation marks (" ") creates a vertical row

For example:
"Dear customer *xyz*,"
"Enclosed is a check for 1000.00 to be applied as a payment toward account number: 123456789. If you have any questions or need additional information, contact us at (888) 555-*xxxx*. Our business hours are Mon-Fri 8:00am-5:00pm Eastern Time.",
"Sincerely,",
Your lender

After a check is issued, it takes about 5 - 7 business days for receipt (CRB uses USPS first class mail). If a check isn't cashed after 15 calendar days, it is canceled.

You can contact MPL Ops to stop payment on a check, and for any questions regarding check delivery or receipt.

Webhooks are reported when a check is processed, mailed, cashed or expired. For more information on webhooks, see View webhook events available.

> 📘
>
> ### Important
>
> When you add a date field in a request, the format must be: **yyyy-mm-dd**, such as 1970-12-05.

```
{
{
  "FundingInfoUpdate": {
   "Rails": [
         {
    "CheckFields": {
          "PhysicalAddressOfRecipient": {
            "Line_1": "Anita Loan",
            "Line_2": "Strawberry Fields Lane",
            "City": "Paradise City",
            "State": "NY",
            "Zip": "11252",
            "Country": "USA"
          },
          "Name": "Anita",
          "Attachment": "{Encoded64 text here}",
          "Description": "Loan from Cross River",
          "Remittance_advice": "",
          "IsPrintedandMailed": "0",
          "CheckEmail": "anita@gmail.com"
        },
"RailType":"CHECK",
"Priority":1,
"Amount":200.0
}
    ]
  }
},
```

Attribute	Description
Rails *array*	Rail selected, in this case **Check**.
Line_1 *string*	Check delivery address, line 1. **Required**
Line_2 *string*	Check delivery address, line 2. *Optional*
City *string*	City of business for check delivery. **Required**
State *string*	State of business for check delivery. 2 letters **Required**
Zip *string*	Zip code of business for check delivery. **Required**
Country *string*	Recipient's country. *Always US* 2-letter country code. *Optional*
Name *string*	Name of the borrower. *Optional*
Description *string*	The name of the loan being paid. This is printed on the check and looks best if under 100 chars. *Optional*
Attachment *string*	Base64 encoded 1 page PDF. It must be encoded and inserted here as text. *Optional*
Remittance_advice *array*	Base64 encoded text. Up to 700 chars before encoding. Appears on the top third of the page. The check is located on the bottom third of the page. *Optional*
IsPrintedandMailed *boolean*	Whether the the check is being printed and mailed. True if check is being printed and mailed, otherwise, false. **Required**
CheckEmail *string*	Recipient's email. *Optional*
Priority *integer*	Rail priority. **Required**
Amount *integer*	Amount being funded. **Required**

Check parameters

To pay down loan proceeds, you can send a physical check to a biller (merchant or business). This is useful when you have to pay billers that don't accept payments via Mastercard RPPS.

You can add an optional Base64 encoded attachment to accompany the check, and it will be mailed as a PDF.

To format the field **Remittance_advice** so that it reads correctly, follow the details points below:

- Quotation marks (" ") creates a vertical row

For example:
"Dear customer *xyz*,"
"Enclosed is a check for 1000.00 to be applied as a payment toward account number: 123456789. If you have any questions or need additional information, contact us at (888) 555-*xxxx*. Our business hours are Mon-Fri 8:00am-5:00pm Eastern Time.",
"Sincerely,",
Your lender

After a check is issued, it takes about 5 - 7 business days for receipt (CRB uses USPS first class mail). If a check isn't cashed after 15 calendar days, it is canceled.

You can contact MPL Ops to stop payment on a check, and for any questions regarding check delivery or receipt.

Webhooks are reported when a check is processed, mailed, cashed or expired. For more information on webhooks, see View webhook events available.

> 📘
>
> ### Important
>
> When you add a date field in a request, the format must be: **yyyy-mm-dd**, such as 1970-12-05.

```
{
{
  "FundingInfoUpdate": {
   "Rails": [
         {
    "CheckFields": {
          "PhysicalAddressOfRecipient": {
            "Line_1": "Anita Loan",
            "Line_2": "Strawberry Fields Lane",
            "City": "Paradise City",
            "State": "NY",
            "Zip": "11252",
            "Country": "USA"
          },
          "Name": "Anita",
          "Attachment": "{Encoded64 text here}",
          "Description": "Loan from Cross River",
          "Remittance_advice": "",
          "IsPrintedandMailed": "0",
          "CheckEmail": "anita@gmail.com"
        },
"RailType":"CHECK",
"Priority":1,
"Amount":200.0
}
    ]
  }
},
```

Attribute	Description
Rails *array*	Rail selected, in this case **Check**.
Line_1 *string*	Check delivery address, line 1. **Required**
Line_2 *string*	Check delivery address, line 2. *Optional*
City *string*	City of business for check delivery. **Required**
State *string*	State of business for check delivery. 2 letters **Required**
Zip *string*	Zip code of business for check delivery. **Required**
Country *string*	Recipient's country. *Always US* 2-letter country code. *Optional*
Name *string*	Name of the borrower. *Optional*
Description *string*	The name of the loan being paid. This is printed on the check and looks best if under 100 chars. *Optional*
Attachment *string*	Base64 encoded 1 page PDF. It must be encoded and inserted here as text. *Optional*
Remittance_advice *array*	Base64 encoded text. Up to 700 chars before encoding. Appears on the top third of the page. The check is located on the bottom third of the page. *Optional*
IsPrintedandMailed *boolean*	Whether the the check is being printed and mailed. True if check is being printed and mailed, otherwise, false. **Required**
CheckEmail *string*	Recipient's email. *Optional*
Priority *integer*	Rail priority. **Required**
Amount *integer*	Amount being funded. **Required**

XPay parameters

XPay transfers funds between CRB and an MPL account within the Cross River Operating System (COS). XPay transfers are executed internally within CRB and are immediate. Once funds are in the MPL account, they can use the funds in any of these use cases:

- Fund 3rd parties to make the final payment to the borrower
- ACH to merchants (Point-of-Sale payments)
- Recycle a loan for a repeat customer


```
{
    "LoanId": "bbedd717-8589-4411-b6da-ac6e0097bf51",
    "Rails": [
        {
        "XPayFields": {
            "ToAccountName": "",
            "ToAccountNumber": "2392433617",
            "Description": "A1B2C3"
            },
            "RailType": "COSXPay",
            "Priority": 1,
            "Amount": 7111.12
      }
    ]
}
```


Attribute	Description
ToAccountNumber *string*	The unique identifier of the receiver. This defaults to the COS account number. **Required**
Description *string*	A description of the transaction to be used by the partner to identify the transaction. Minimum length of 1, maximum length of 30. **Required**
ToAccountName *string*	Name of the account receiving the funds. **Required**

ACH parameters

Arix uses ACH to push money into the account of a borrower, vendor, or creditor. CRB must approve every ACH payment that is sent.

There are two types of ACH rails:

- **COSACH**
  - Via NACHA
  - Funded next day
- **COSACHSD**
  - Via NACHA
  - Same day funding

JSON

```
{
    "LoanId": "bbedd717-8589-4411-b6da-ac6e0097bf51",
    "Rails": [
        {
            "ACHFields": {
                "StandardEntryClassCode": "PPD",
                "TransactionType": "push",
                "ToRoutingNumber": "021214891",
                "ToAccountType": "Checking",
                "ToAccountName": "Anita Lon",
                "ToAccountNumber": "9999999999",
                "Description": "Loan"
            },
            "Id": "23b5fedb-bc20-41a0-a4ae-ac9300b4530e",
            "RailType": "COSACH",
            "Priority": 1,
            "Amount": 100
        }
```

Request parameters

Attributes	Description
ToRoutingNumber *string*	Receiving bank's routing number. The value entered for this field is validated up front and it must be 9 digits long. **Required**
ToAccountName *string*	Name on the customer account. Limited to 22 characters. **Required**
ToAccountNumber *string*	Account number. Limited to 17 digits. **Required**
Description *string*	Mapped to identification field in the NACHA file. Limited to 15 characters. **Required**
StandardEntryClassCode *string*	The type of account: - PPD - consumer accounts - CCD - corporate accounts **Required**
TransactionType *string*	Push. **Required**
ToAccountType *string*	Checking account or savings account. **Required**

Updated about 2 months ago

← Payment rails

ACH returns →

Did this page help you? Yes No

ACH returns

ACH credits aren't final as are wires. When Arix uses an ACH rail to process a loan, the rail can be returned from the Fed at any point in the loan process. Returns typically occur within several days of funding, but they can be generated up to 180 days from the funding date. Click here for more information.

Arix uses webhooks to update the funding status of a loan.

What happens in Arix when a rail is returned

When a rail is returned, the status of the rail is automatically updated to **Returned**.

The loan status changes based on these conditions:

- **Returned**
  If all rails were returned, then the loan automatically moves into the status, **Returned**, regardless of the previous loan status.

- **NotFullyFunded**
  If at least one rail was returned, the loan moves into the status, **NotFullyFunded**. See the enums reference document for a list of the loan and rail statuses.

These webhooks are triggered for the rail and loan statuses:

- **LoanStatusUpdated** (if the return causes the loan status to change)
- **RailUpdated**

The loan funding information is updated accordingly in these fields:

- **ProcessedRails**
- **ReturnedRails**

Investigating returned rails

You can call GET /LoanDetails to view the details on a returned rail by including returned rails in the field request. The update includes the NACHA return reason code, the return trace number, and the original trace number.

Resubmitting a returned rail

To resubmit a returned rail, call PUT /FundingInfo with the correct details.

Adjustment Action parameters

You can update the loan information after a loan was funded (or partially funded). Update the relevant parameters from the list below. Any loan funding information that you changed, shows with the word **Adjusted** before it.

Attribute	Description
AdjustmentActionType *LoanAdjustmentAction*	Type of update made to the loan. **Required**
ActionAmount *decimal*	Dollar amount of the change to the loan. **Required**
Description *string*	Description of the reason for the loan adjustment. 1 - 30 characters. **Required**
UpdatedBy *string*	By whom the information was updated. *Optional*
UpdatedOn *string*	Date/time the information was updated. *Optional*
LoanId *string*	Unique identifier of the loan. *Optional*
AdjustedLoanAmount *decimal or null*	Adjusted gross dollar amount requested. *Optional*
AdjustedNetFunding *decimal or null*	Adjusted net dollar amount to fund. *Optional*
AdjustedOriginationFee *decimal or null*	Adjusted charge to the borrower for the loan. *Optional*
AdjustedCurrentMonthlyPayment *decimal or null*	Adjusted monthly payment amount. *Optional*
AdjustedMerchantFee *decimal or null*	Adjusted merchant fee. *Optional*
AdjustedFinalPaymentDate *DateTime or null*	Adjusted date due for final loan payment. The format must be: **yyyy-mm-dd**. *Optional*
AdjustedFinalPayment *decimal or null*	Adjusted dollar amount of final payment. *Optional*
AdjustedFinanceCharge *decimal or null*	Adjusted finance charge paid by the borrower. *Optional*
AdjustedTotalPayments *decimal or null*	Adjusted total paid by the borrower after all the payments have been made. *Optional*
AdjustedAPR *decimal or null*	Adjusted annual percentage interest rate, as a percent. *Optional*

Updated 10 months ago

← ACH returns

Check parameters →

Did this page help you? Yes No

Check parameters

To pay down loan proceeds, you can send a physical check to a biller (merchant or business). This is useful when you have to pay billers that don't accept payments via Mastercard RPPS.

You can add an optional Base64 encoded attachment to accompany the check, and it will be mailed as a PDF.

To format the field **Remittance_advice** so that it reads correctly, follow the details points below:

- Quotation marks (" ") creates a vertical row

For example:
"Dear customer *xyz*,"
"Enclosed is a check for 1000.00 to be applied as a payment toward account number: 123456789. If you have any questions or need additional information, contact us at (888) 555-*xxxx*. Our business hours are Mon-Fri 8:00am-5:00pm Eastern Time.",
"Sincerely,",
Your lender

After a check is issued, it takes about 5 - 7 business days for receipt (CRB uses USPS first class mail). If a check isn't cashed after 15 calendar days, it is canceled.

You can contact MPL Ops to stop payment on a check, and for any questions regarding check delivery or receipt.

Webhooks are reported when a check is processed, mailed, cashed or expired. For more information on webhooks, see View webhook events available.

> 📘
>
> ### Important
>
> When you add a date field in a request, the format must be: **yyyy-mm-dd**, such as 1970-12-05.

```
{
{
  "FundingInfoUpdate": {
   "Rails": [
         {
    "CheckFields": {
          "PhysicalAddressOfRecipient": {
            "Line_1": "Anita Loan",
            "Line_2": "Strawberry Fields Lane",
            "City": "Paradise City",
            "State": "NY",
            "Zip": "11252",
            "Country": "USA"
          },
          "Name": "Anita",
          "Attachment": "{Encoded64 text here}",
          "Description": "Loan from Cross River",
          "Remittance_advice": "",
          "IsPrintedandMailed": "0",
          "CheckEmail": "anita@gmail.com"
        },
"RailType":"CHECK",
"Priority":1,
"Amount":200.0
}
    ]
  }
},
```

Attribute	Description
Rails *array*	Rail selected, in this case **Check**.
Line_1 *string*	Check delivery address, line 1. **Required**
Line_2 *string*	Check delivery address, line 2. *Optional*
City *string*	City of business for check delivery. **Required**
State *string*	State of business for check delivery. 2 letters **Required**
Zip *string*	Zip code of business for check delivery. **Required**
Country *string*	Recipient's country. *Always US* 2-letter country code. *Optional*
Name *string*	Name of the borrower. *Optional*
Description *string*	The name of the loan being paid. This is printed on the check and looks best if under 100 chars. *Optional*
Attachment *string*	Base64 encoded 1 page PDF. It must be encoded and inserted here as text. *Optional*
Remittance_advice *array*	Base64 encoded text. Up to 700 chars before encoding. Appears on the top third of the page. The check is located on the bottom third of the page. *Optional*
IsPrintedandMailed *boolean*	Whether the the check is being printed and mailed. True if check is being printed and mailed, otherwise, false. **Required**
CheckEmail *string*	Recipient's email. *Optional*
Priority *integer*	Rail priority. **Required**
Amount *integer*	Amount being funded. **Required**

RPPS parameters

You can use the RPPS rail in Arix to electronically transfer funds to pay loan proceeds directly to billers. Borrowers who select to use this rail, can have funds sent directly to billers to pay down their outstanding bills, most typically used to pay down outstanding credit card bills.

The rail uses Mastercard's RPPS rail system. Arix sends an API call to the CRB RPPS gateway, which in turn sends a request to Mastercard to transfer funds. Mastercard then sends a credit to the biller.

RPPS requests are batched daily by the CRB RPPS gateway when the system sends the batched requests to Mastercard for processing.

> 👍
>
> **Note**
>
> The borrower's credit card doesn't have to be a Mastercard. It can be any card issued by a biller that signed up to receive bill payments via the RPPS service.


```
{
    "LoanId": "bbedd717-8589-4411-b6da-ac6e0097bf51",
    "Rails": [
        {
            "RPPSFields": {
                "BillerID": "132654654632135423452",
                "BillerName": "Anita Loan",
                "ConsumerNameAtBiller": "TestNameAtBiller",
                "ConsumerAccountNumberAtBiller": "5454 54XX XXXX 5454 "
            },
            "Id": "11e6f574-0bba-4597-9264-ac9300b2f39a",
            "RailType": "RPPS",
            "Priority": 1,
            "Amount": 7000,
            "LoanId": "bbedd717-8589-4411-b6da-ac6e0097bf51",
            "MPLId": "TST",
            "IsFailed": false,
            "LoanType": "Undefined"
        }
    ]
}
```


Attribute	Description
BillerID *string*	The 10 character ID number of the credit card issuer (biller). If length(BillerID) < 10, you can pad with zeroes. **Example:** *0000*12345 **Required**
ConsumerNameAtBiller *string*	The borrower's cardholder name on the credit card account. **Required**
ConsumerAccountNumberAtBiller *string*	The borrower's credit card account number on the biller's credit card. **Required**
BillerName *string*	The name of the credit card company that is making the disbursement. *Optional*

You can simulate rails in the sandbox. The table indicates the responses that you should receive for the transaction amounts listed.

Amount	Result
Up to $3	Result is `submitted`
Above $3 to $4	Result is `rejected`
Above $4 to $5	Result is `approved` After 30 seconds it becomes `returned`
Above $5	Result is `approved`

Payment Card Industry (PCI) standards ensure that sensitive credit card information is protected from fraudulent use. Arix processing complies with PCI standards.

When an RPPS transaction is initiated, Arix receives a request that includes the borrower's credit card number. This number is sent to a tokenization service that is Level 1 PCI-compliant. They send a token back to Arix, which stores the token.

When processing the RPPS batch, Arix calls to detokenize all the credit card numbers, and sends the credit card numbers in a file to Mastercard.

Arix logs contain only masked credit card numbers.

RPPS returns

RPPS rails are approved immediately after funding and the rail is marked **Success**. If however, an RPPS is returned, these are the guidelines:

- If, within 5 days of the rail attempt RPPS returns a result of **Rejected** or **Failed**, Arix marks the rail as **Rejected**.
- If, within 5 days of the rail attempt, RPPS returns a result of **Returned**, Arix marks the rail as **Returned**.
- If 5 days pass without a **Reject**, **Failed** or **Returned** result from RPPS, Arix marks the rail as **Success**.

Updated 10 months ago

RPPS parameters

RPPS transaction reversals

Did this page help you? Yes No

RPPS transaction reversals

When requesting a reversal of an RPPS transaction, follow this process:

1. Check whether the biller allows reversal using the **reqAdndaRevm** tag that you can find in the XML biller directory.
2. If the biller allows reversal, then send an email to **iprops@crossriver.com** and **mplops@crossriverbank.com** as follows:

- Subject: RPPS REVERSAL REQUEST
- Content:
  - Biller ID
  - Account with biller (first 6 digits & last 4 digits)
  - Name on account
  - Reversal amount

Note: By MC RPPS network rules, the reversal request must come in no later than 5 business days from the original payment. Also note that Mastercard has established debit caps that limit the daily amount of reversals. The amount is usually low ($2,000).

Updated 10 months ago

← RPPS returns

XPay parameters →

Did this page help you? Yes No

XPay parameters

XPay transfers funds between CRB and an MPL account within the Cross River Operating System (COS). XPay transfers are executed internally within CRB and are immediate. Once funds are in the MPL account, they can use the funds in any of these use cases:

- Fund 3rd parties to make the final payment to the borrower
- ACH to merchants (Point-of-Sale payments)
- Recycle a loan for a repeat customer


```
{
    "LoanId": "bbedd717-8589-4411-b6da-ac6e0097bf51",
    "Rails": [
        {
        "XPayFields": {
            "ToAccountName": "",
            "ToAccountNumber": "2392433617",
            "Description": "A1B2C3"
            },
            "RailType": "COSXPay",
            "Priority": 1,
            "Amount": 7111.12
      }
    ]
}
```


Attribute	Description
ToAccountNumber *string*	The unique identifier of the receiver. This defaults to the COS account number. **Required**
Description *string*	A description of the transaction to be used by the partner to identify the transaction. Minimum length of 1, maximum length of 30. **Required**
ToAccountName *string*	Name of the account receiving the funds. **Required**

Loan Reports

CRB requires additional data from an MPL. The data is sent via SFTP files (tapes). To transfer the files securely, you must use a client that supports SSH with the following public-key algorithms:

- RSA
- sha2 521
- sha2 384
- sha 256

These loan reports are available:

- Application
- Servicing
- Transaction

Updated about 1 month ago

XPay parameters

Application data exchange

Did this page help you? Yes No

Application data exchange

An MPL sends CRB a daily report/file - the Application Tape - to enable CRB to perform a fair lending review. The report contains the applications from the previous day that were decisioned by the MPL; those that were **Approved** and sent to Arix for origination and funding, and those that were **Declined**.

API is coming

CRB will be replacing the file based data exchange with an API in a future release.

Swagger: https://lendingpreapprovalstg.crbnj.net/swagger/index.html
API calls:

1. POST /v2/PreApproval/Approve
2. POST /v2/PreApproval/deny

The table below is the reference information for applications submitted.

Notes:

- If a loan changes status (such as from denied to funded), the updated loan record should be included on the tape with the **UpdatedFlag**. (A flag indicates that this record updates a previous credit decision for this loan).
- During implementation, CRB and the MPL will agree on additional fields to be passed to CRB based on the MPL's unique credit model.

- File format is CSV.
- Naming convention: MplId_ApplicationTape_yyyymmdd.csv, where yyyymmdd is the date of file generation.
- Data in the report should be prior day applications.

Attribute	Action	Date
ProgramType MarketingChannel CreditPolicyVersion CreditModelVersion InvestorName InvestorDecisioned UnderwriterNotes DenialReason PolicyDeclineReason VantageScore VantageScoreDate NetDisposableIncome PreDTIExMortgage PreDTIInMortgage PreDTIDate PostDTIDate PostDTIExMortgage PostDTIInMortgage	Added	June 16, 2021
DenialReason 1 - 10 PolicyDeclineReason 1-10 DTI	Consolidated Expanded	June 16, 2021

Attribute	Description
BorrowerId *alphanumeric*	Unique identifier for the borrower.
ApplicationId *alphanumeric*	Unique identifier for the application.
LoanNumber *alphanumeric*	Unique identifier for the loan.
Platform *text*	MPL name.
ProgramType *text*	MPL defined program to which the loan belongs Same as the program fields sent to Arix.

MarketingChannel *text*	Marketing channel name and code as defined by the credit policy.
CreditPolicyVersion *alphanumeric*	Credit policy version.
CreditModelVersion *alphanumeric*	Credit model version.
InvestorName *text*	Name of the investor that decisioned this loan, if applicable.
InvestorDecisioned 1 or 0	1 if the investor decision overrode the MPL decision, otherwise 0.
LoanPurpose *text*	Borrower-stated loan purpose.
AssetClass *integer*	1 - Student 2 - Solar 3 - Debt consolidation 4 - Home improvement 5 - POS 6 - Medical 7 - Auto loan 99 - Other
FinalActionTaken *text*	Decisioning: **Approved**, **Denied**, **Incomplete**, or **Counteroffer** - **Approved** - the application was approved, the offer was accepted and approved by CRB, or it hasn't yet passed the offer stage. - **Denied** - if the application is declined at credit checks. - **Counteroffer** - if a counteroffer is sent to the borrower. - **Counteroffer accepted** - if a counteroffer was sent to a borrower and the borrower accepted. - **Counteroffer not accepted** - if a counteroffer was sent to a borrower and the borrower declined. - **Approved not accepted** - if the application was approved but the offer wasn't accepted. - **CRB denied** - if the offer was accepted and then rejected by CRB.

	- **Incomplete** - if the borrower didn't complete the application.
UnderwriterNotes *text*	Free text field for underwriter notes.
DenialReason *json blob*	json blob of all denial reasons. The blob must EXCLUDE text delimiters and line feeds. ""DenialReasons"": { ""DenialReason1"": ""denial_reason_1"", ""DenialReason2"": ""denial_reason_2"" }" **See note below for explanation**
PolicyDeclineReason *json blob*	Decline reason per credit policy (DTI, FICO, etc.) json blob of all policy decline reasons. The blob must EXCLUDE text delimiters and line feeds. ""PolicyDeclineReasons"": { ""PolicyDeclineReason1"": ""policy_decline_reason_1"", ""PolicyDeclineReason2"": ""policy_decline_reason_2"" }" **See note below for explanation.**
DenialDate *date*	Date the application was denied.
Application Date *date*	Date the consumer applied for the loan.
Reg B Decision Date *date*	Date decisioning occurred.
AdverseActionNoticeDate *date*	Date the MPL emailed/mailed the adverse action notice to the applicant. **Note:** All adverse action notices must be provided to a consumer within 30 days of a denial of credit.

CounterofferDate *date*	Date the MPL provided a consumer with a counter offer in response to their credit application. **Note**: Counter offers must be presented to an applicant within 30 days of receiving a complete application.
NoticeOfIncompleteness *date*	Date that a notice of incomplete application was sent to the borrower.
InitialFraudAlert Y or N	An initial fraud alert can be placed on a consumer's credit report if they think they were a victim of fraud or identity theft. The alert is found on the credit report and stays on file for 90 days. Yes for an initial fraud alert, otherwise no.
ExtendedFraudAlert Y or N	An extended fraud alert can be placed on a consumer's credit report if they were a victim of identity theft. The alert is found on the credit report and stays on file for 7 years. Yes for an extended fraud alert, otherwise no.
AddressDiscrepancyFlag Y or N	This alert, found on the credit report, indicates that the address provided by a user to obtain a report **substantially differs** from the address the NCRA has in the consumer's file. Yes if there is an address discrepancy, otherwise no.
CreditFreeze Y or N	This alert, found on the credit report, restricts access to the credit report. Yes to restrict access, otherwise no.
ExceptionFlag Y or N	Indicates if there are any exceptions to the policy. Yes if there are exceptions, otherwise no.
ExceptionNotes *text*	Description of the exception detailing why an exception was granted.
BorrowerLastName *text*	Borrower's last name.
BorrowerFirstName *text*	Borrower's first name.
BorrowerDOB *alphanumeric*	Borrower's date of birth.

BorrowerSSN *alphanumeric*	Borrower's Social Security Number.
BorrowerAddress *text*	Borrower's street address. Must be a physical address and not a PO Box.
BorrowerCity *text*	Borrower's city of residence.
BorrowerState *text*	Borrower's State of residence.
BorrowerZip *alphanumeric*	Borrower's 5-digit zip code.
BorrowerPhone *alphanumeric*	Borrower's phone number.
BorrowerEmail *alphanumeric*	Borrower's email address.
RequestedLoanAmount *number to 2 decimals*	Amount the borrower requested at the time of application. The requested amount can change before the final loan amount is settled, for borrower or lender related reasons.
LoanAmount *number to 2 decimals*	Actual gross amount borrowed.
OriginationFee *number to 2 decimals*	Charge to the borrower for the loan per the guidelines.
PeriodicPmt *number to 2 decimals*	Amount of regular payment per the note.
FinalPaymentDate *date*	Date of the last regular payment.
FinalPayment *number to 2 decimals*	Amount of the final payment.
FinanceCharge *number to 2 decimals*	Finance charge paid by the borrower.
TotalPayments *number to 2 decimals*	Total paid by the borrower after all payments are made.

PPY *integer*	Number of payments per year.
Rate *number to 2 decimals*	Interest rate, as a percent.
APR *number to 2 decimals*	Annual percentage rate, as a percent.
Amortization *integer*	Number of payments at origination to pay off the loan.
Term *integer*	Number of months at origination to pay off the loan.
RateType *integer* - 1 or 2	1 - Fixed 2 - Floating
PriorLoanFlag Y or N	Has the borrower had other loans with the sponsor. Yes if the borrower has had other loans with the sponsor, otherwise no.
FICO *integer - 3 digits*	Use the 3-digit FICO score only.
FICODate *date*	Date FICO score was obtained.
VantageScore *text*	Vantage score at application.
VantageScoreDate *date*	Date Vantage Score was obtained.
CreditGrade *alphanumeric*	If applicable, internal grade for the loan from the credit policy (Risk Group).
DebtUtilization *number to 2 decimals*	Percent of credit used by the borrower.
TotalRevolvingDebt *number to 2 decimals*	Dollar amount of all revolving debt on the credit report.
CreditInquiries12months *integer*	Number of credit report inquiries in the last 12 months.

DQPast24Months *integer*	Number of times the borrower submitted late payments in the past 24 months on all accounts, per the credit report.
AccountsOpenedPast24Months *integer*	Number of accounts opened by the borrower in the past 24 months, per the credit report.
OpenCreditLines *integer*	Number of open trade lines on the credit report.
CollectionsExcludingMedical *integer*	Number of open collection accounts per the credit report, not including medical.
PublicRecordsOnFile *number of negative public records*	Number of open public records shown on the credit report such as bankruptcies, tax liens and judgements.
MonthsSinceLastRecord *integer*	Number of months since the borrower's last public record.
HomeownerFlag Y or N	If the borrower is already a homeowner - use multiple sources to determine, including the application. Yes if the borrower is a homeowner, otherwise no.
EmploymentLength *integer*	Number of months the borrower has been employed at their current job.
EmployementLengthDescription *text*	Number of years the borrower has been employed at their current job.
Employer *text*	Employer's name.
AnnualIncome *number up to 2 decimals*	Dollar amount of annual income.
NetDisposableIncome *number up to 2 decimals*	Net income remaining after debt (as defined by credit policy, if applicable).
PreDTIExMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI excluding mortgage pre-loan **as a percentage**.
PreDTIInMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI including mortgage pre-loan **as a percentage**.
PreDTIDate *number up to 2 decimals*	**Debt to Income Ratio** DTI date pulled pre-loan **as a percentage**.

PostDTIDate *number up to 2 decimals*	**Debt to Income Ratio** DTI date pulled post-loan **as a percentage**.
PostDTIExMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI excluding Mortgage post-loan **as a percentage**.
PostDTIInMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI including Mortgage post-loan **as a percentage**.
IPAddress xxx.xxx.xxx.xxx	IP address of the computer the borrower applied on.
MLAFLAG Y or N	If the loan is being made to a covered borrower. Yes if the borrower is covered, otherwise No.
MAPR *number up to 2 decimals*	Military APRA standard calculation used by lenders for loans covered by the Department of Defense's Military Lending Act & Regulation Z. MAPR is required if MLAFLAG = Yes For more information, see: https://www.fdic.gov/regulations/compliance/manual/5/V-13.1.pdf
UpdatedFlag 1 or 0	Flag indicates that this record updates a previous credit decision for this loan. 1 - True - Updates a previous credit decision. 0 - False - Doesn't update a previous credit decision.
MerchantName *text*	Name of merchant, if applicable.
MerchantFee *number to 2 decimals*	Amount paid to merchant, if applicable.
CoBorrowerLastName *text*	CoBorrower's last name, if applicable.
CoBorrowerFirstName *text*	CoBorrower's first name, if applicable.
CoBorrowerAddress *alphanumeric*	CoBorrower's address. Must be a physical address and not a PO Box.

CoBorrowerCity *alphanumeric*	CoBorrower's city of residence
CoBorrowerState *text*	CoBorrower's State of residence.
CoBorrowerZip *text*	CoBorrower's zip code.
CoBorrowerDOB *text*	CoBorrower's date of birth.
CoBorrowerSSN *alphanumeric*	CoBorrower's Social Security Number.
CoBorrowerPhone *alphanumeric*	CoBorrower's phone number.
CoBorrowerEmail *alphanumeric*	CoBorrower's email address.
CoBorrowerFICO *numeric*	CoBorrower's FICO score.
ExtraFields *json blob*	json blob allowing for all extra fields. The blob must EXCLUDE text delimiters and line feeds. Any changes/additions to this field must go through change management.

Note: **DenialReason** versus **PolicyDeclineReason**

The difference between DenialReason and PolicyDeclineReason is a conceptual one:

- The DeclineReason should be the one stated in the AAN.
- The PolicyDeclineReason is the specific rule in the underwriting policy that the applicant didn't satisfy.

  Most of the time they match, but there can be instances when the AAN reason is worded differently than the PolicyDeclineReason.

Example: The AAN (DenialReason) stated "Our partner bank can't approve the loan you requested because we couldn't verify your identity based on the information we received". The reason stated in the PolicyDeclineReason however, is that the applicant was found on OFAC.

Application data exchange

An MPL sends CRB a daily report/file - the Application Tape - to enable CRB to perform a fair lending review. The report contains the applications from the previous day that were decisioned by the MPL; those that were **Approved** and sent to Arix for origination and funding, and those that were **Declined**.

API is coming

CRB will be replacing the file based data exchange with an API in a future release.

Swagger: https://lendingpreapprovalstg.crbnj.net/swagger/index.html
API calls:

1. POST /v2/PreApproval/Approve
2. POST /v2/PreApproval/deny

The table below is the reference information for applications submitted.

Notes:

- If a loan changes status (such as from denied to funded), the updated loan record should be included on the tape with the **UpdatedFlag**. (A flag indicates that this record updates a previous credit decision for this loan).
- During implementation, CRB and the MPL will agree on additional fields to be passed to CRB based on the MPL's unique credit model.

- File format is CSV.
- Naming convention: MplId_ApplicationTape_yyyymmdd.csv, where yyyymmdd is the date of file generation.
- Data in the report should be prior day applications.

Attribute	Action	Date
ProgramType MarketingChannel CreditPolicyVersion CreditModelVersion InvestorName InvestorDecisioned UnderwriterNotes DenialReason PolicyDeclineReason VantageScore VantageScoreDate NetDisposableIncome PreDTIExMortgage PreDTIInMortgage PreDTIDate PostDTIDate PostDTIExMortgage PostDTIInMortgage	Added	June 16, 2021
DenialReason 1 - 10 PolicyDeclineReason 1-10 DTI	Consolidated Expanded	June 16, 2021

Attribute	Description
BorrowerId *alphanumeric*	Unique identifier for the borrower.
ApplicationId *alphanumeric*	Unique identifier for the application.
LoanNumber *alphanumeric*	Unique identifier for the loan.
Platform *text*	MPL name.
ProgramType *text*	MPL defined program to which the loan belongs Same as the program fields sent to Arix.

MarketingChannel *text*	Marketing channel name and code as defined by the credit policy.
CreditPolicyVersion *alphanumeric*	Credit policy version.
CreditModelVersion *alphanumeric*	Credit model version.
InvestorName *text*	Name of the investor that decisioned this loan, if applicable.
InvestorDecisioned 1 or 0	1 if the investor decision overrode the MPL decision, otherwise 0.
LoanPurpose *text*	Borrower-stated loan purpose.
AssetClass *integer*	1 - Student 2 - Solar 3 - Debt consolidation 4 - Home improvement 5 - POS 6 - Medical 7 - Auto loan 99 - Other
FinalActionTaken *text*	Decisioning: **Approved**, **Denied**, **Incomplete**, or **Counteroffer** - **Approved** - the application was approved, the offer was accepted and approved by CRB, or it hasn't yet passed the offer stage. - **Denied** - if the application is declined at credit checks. - **Counteroffer** - if a counteroffer is sent to the borrower. - **Counteroffer accepted** - if a counteroffer was sent to a borrower and the borrower accepted. - **Counteroffer not accepted** - if a counteroffer was sent to a borrower and the borrower declined. - **Approved not accepted** - if the application was approved but the offer wasn't accepted. - **CRB denied** - if the offer was accepted and then rejected by CRB.

	- **Incomplete** - if the borrower didn't complete the application.
UnderwriterNotes *text*	Free text field for underwriter notes.
DenialReason *json blob*	json blob of all denial reasons. The blob must EXCLUDE text delimiters and line feeds. ""DenialReasons"": { ""DenialReason1"": ""denial_reason_1"", ""DenialReason2"": ""denial_reason_2"" }" **See note below for explanation**
PolicyDeclineReason *json blob*	Decline reason per credit policy (DTI, FICO, etc.) json blob of all policy decline reasons. The blob must EXCLUDE text delimiters and line feeds. ""PolicyDeclineReasons"": { ""PolicyDeclineReason1"": ""policy_decline_reason_1"", ""PolicyDeclineReason2"": ""policy_decline_reason_2"" }" **See note below for explanation.**
DenialDate *date*	Date the application was denied.
Application Date *date*	Date the consumer applied for the loan.
Reg B Decision Date *date*	Date decisioning occurred.
AdverseActionNoticeDate *date*	Date the MPL emailed/mailed the adverse action notice to the applicant. **Note:** All adverse action notices must be provided to a consumer within 30 days of a denial of credit.

CounterofferDate *date*	Date the MPL provided a consumer with a counter offer in response to their credit application. **Note**: Counter offers must be presented to an applicant within 30 days of receiving a complete application.
NoticeOfIncompleteness *date*	Date that a notice of incomplete application was sent to the borrower.
InitialFraudAlert Y or N	An initial fraud alert can be placed on a consumer's credit report if they think they were a victim of fraud or identity theft. The alert is found on the credit report and stays on file for 90 days. Yes for an initial fraud alert, otherwise no.
ExtendedFraudAlert Y or N	An extended fraud alert can be placed on a consumer's credit report if they were a victim of identity theft. The alert is found on the credit report and stays on file for 7 years. Yes for an extended fraud alert, otherwise no.
AddressDiscrepancyFlag Y or N	This alert, found on the credit report, indicates that the address provided by a user to obtain a report **substantially differs** from the address the NCRA has in the consumer's file. Yes if there is an address discrepancy, otherwise no.
CreditFreeze Y or N	This alert, found on the credit report, restricts access to the credit report. Yes to restrict access, otherwise no.
ExceptionFlag Y or N	Indicates if there are any exceptions to the policy. Yes if there are exceptions, otherwise no.
ExceptionNotes *text*	Description of the exception detailing why an exception was granted.
BorrowerLastName *text*	Borrower's last name.
BorrowerFirstName *text*	Borrower's first name.
BorrowerDOB *alphanumeric*	Borrower's date of birth.

BorrowerSSN *alphanumeric*	Borrower's Social Security Number.
BorrowerAddress *text*	Borrower's street address. Must be a physical address and not a PO Box.
BorrowerCity *text*	Borrower's city of residence.
BorrowerState *text*	Borrower's State of residence.
BorrowerZip *alphanumeric*	Borrower's 5-digit zip code.
BorrowerPhone *alphanumeric*	Borrower's phone number.
BorrowerEmail *alphanumeric*	Borrower's email address.
RequestedLoanAmount *number to 2 decimals*	Amount the borrower requested at the time of application. The requested amount can change before the final loan amount is settled, for borrower or lender related reasons.
LoanAmount *number to 2 decimals*	Actual gross amount borrowed.
OriginationFee *number to 2 decimals*	Charge to the borrower for the loan per the guidelines.
PeriodicPmt *number to 2 decimals*	Amount of regular payment per the note.
FinalPaymentDate *date*	Date of the last regular payment.
FinalPayment *number to 2 decimals*	Amount of the final payment.
FinanceCharge *number to 2 decimals*	Finance charge paid by the borrower.
TotalPayments *number to 2 decimals*	Total paid by the borrower after all payments are made.

PPY *integer*	Number of payments per year.
Rate *number to 2 decimals*	Interest rate, as a percent.
APR *number to 2 decimals*	Annual percentage rate, as a percent.
Amortization *integer*	Number of payments at origination to pay off the loan.
Term *integer*	Number of months at origination to pay off the loan.
RateType *integer* - 1 or 2	1 - Fixed 2 - Floating
PriorLoanFlag Y or N	Has the borrower had other loans with the sponsor. Yes if the borrower has had other loans with the sponsor, otherwise no.
FICO *integer - 3 digits*	Use the 3-digit FICO score only.
FICODate *date*	Date FICO score was obtained.
VantageScore *text*	Vantage score at application.
VantageScoreDate *date*	Date Vantage Score was obtained.
CreditGrade *alphanumeric*	If applicable, internal grade for the loan from the credit policy (Risk Group).
DebtUtilization *number to 2 decimals*	Percent of credit used by the borrower.
TotalRevolvingDebt *number to 2 decimals*	Dollar amount of all revolving debt on the credit report.
CreditInquiries12months *integer*	Number of credit report inquiries in the last 12 months.

DQPast24Months *integer*	Number of times the borrower submitted late payments in the past 24 months on all accounts, per the credit report.
AccountsOpenedPast24Months *integer*	Number of accounts opened by the borrower in the past 24 months, per the credit report.
OpenCreditLines *integer*	Number of open trade lines on the credit report.
CollectionsExcludingMedical *integer*	Number of open collection accounts per the credit report, not including medical.
PublicRecordsOnFile *number of negative public records*	Number of open public records shown on the credit report such as bankruptcies, tax liens and judgements.
MonthsSinceLastRecord *integer*	Number of months since the borrower's last public record.
HomeownerFlag Y or N	If the borrower is already a homeowner - use multiple sources to determine, including the application. Yes if the borrower is a homeowner, otherwise no.
EmploymentLength *integer*	Number of months the borrower has been employed at their current job.
EmployementLengthDescription *text*	Number of years the borrower has been employed at their current job.
Employer *text*	Employer's name.
AnnualIncome *number up to 2 decimals*	Dollar amount of annual income.
NetDisposableIncome *number up to 2 decimals*	Net income remaining after debt (as defined by credit policy, if applicable).
PreDTIExMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI excluding mortgage pre-loan **as a percentage**.
PreDTIInMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI including mortgage pre-loan **as a percentage**.
PreDTIDate *number up to 2 decimals*	**Debt to Income Ratio** DTI date pulled pre-loan **as a percentage**.

PostDTIDate *number up to 2 decimals*	**Debt to Income Ratio** DTI date pulled post-loan **as a percentage**.
PostDTIExMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI excluding Mortgage post-loan **as a percentage**.
PostDTIInMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI including Mortgage post-loan **as a percentage**.
IPAddress xxx.xxx.xxx.xxx	IP address of the computer the borrower applied on.
MLAFLAG Y or N	If the loan is being made to a covered borrower. Yes if the borrower is covered, otherwise No.
MAPR *number up to 2 decimals*	Military APRA standard calculation used by lenders for loans covered by the Department of Defense's Military Lending Act & Regulation Z. MAPR is required if MLAFLAG = Yes For more information, see: https://www.fdic.gov/regulations/compliance/manual/5/V-13.1.pdf
UpdatedFlag 1 or 0	Flag indicates that this record updates a previous credit decision for this loan. 1 - True - Updates a previous credit decision. 0 - False - Doesn't update a previous credit decision.
MerchantName *text*	Name of merchant, if applicable.
MerchantFee *number to 2 decimals*	Amount paid to merchant, if applicable.
CoBorrowerLastName *text*	CoBorrower's last name, if applicable.
CoBorrowerFirstName *text*	CoBorrower's first name, if applicable.
CoBorrowerAddress *alphanumeric*	CoBorrower's address. Must be a physical address and not a PO Box.

CoBorrowerCity *alphanumeric*	CoBorrower's city of residence
CoBorrowerState *text*	CoBorrower's State of residence.
CoBorrowerZip *text*	CoBorrower's zip code.
CoBorrowerDOB *text*	CoBorrower's date of birth.
CoBorrowerSSN *alphanumeric*	CoBorrower's Social Security Number.
CoBorrowerPhone *alphanumeric*	CoBorrower's phone number.
CoBorrowerEmail *alphanumeric*	CoBorrower's email address.
CoBorrowerFICO *numeric*	CoBorrower's FICO score.
ExtraFields *json blob*	json blob allowing for all extra fields. The blob must EXCLUDE text delimiters and line feeds. Any changes/additions to this field must go through change management.

Note: **DenialReason** versus **PolicyDeclineReason**

The difference between DenialReason and PolicyDeclineReason is a conceptual one:

- The DeclineReason should be the one stated in the AAN.
- The PolicyDeclineReason is the specific rule in the underwriting policy that the applicant didn't satisfy.

  Most of the time they match, but there can be instances when the AAN reason is worded differently than the PolicyDeclineReason.

Example: The AAN (DenialReason) stated "Our partner bank can't approve the loan you requested because we couldn't verify your identity based on the information we received". The reason stated in the PolicyDeclineReason however, is that the applicant was found on OFAC.

Application data exchange

An MPL sends CRB a daily report/file - the Application Tape - to enable CRB to perform a fair lending review. The report contains the applications from the previous day that were decisioned by the MPL; those that were **Approved** and sent to Arix for origination and funding, and those that were **Declined**.

API is coming

CRB will be replacing the file based data exchange with an API in a future release.

Swagger: https://lendingpreapprovalstg.crbnj.net/swagger/index.html
API calls:

1. POST /v2/PreApproval/Approve
2. POST /v2/PreApproval/deny

The table below is the reference information for applications submitted.

Notes:

- If a loan changes status (such as from denied to funded), the updated loan record should be included on the tape with the **UpdatedFlag**. (A flag indicates that this record updates a previous credit decision for this loan).
- During implementation, CRB and the MPL will agree on additional fields to be passed to CRB based on the MPL's unique credit model.

- File format is CSV.
- Naming convention: MplId_ApplicationTape_yyyymmdd.csv, where yyyymmdd is the date of file generation.
- Data in the report should be prior day applications.

Attribute	Action	Date
ProgramType MarketingChannel CreditPolicyVersion CreditModelVersion InvestorName InvestorDecisioned UnderwriterNotes DenialReason PolicyDeclineReason VantageScore VantageScoreDate NetDisposableIncome PreDTIExMortgage PreDTIInMortgage PreDTIDate PostDTIDate PostDTIExMortgage PostDTIInMortgage	Added	June 16, 2021
DenialReason 1 - 10 PolicyDeclineReason 1-10 DTI	Consolidated Expanded	June 16, 2021

Attribute	Description
BorrowerId *alphanumeric*	Unique identifier for the borrower.
ApplicationId *alphanumeric*	Unique identifier for the application.
LoanNumber *alphanumeric*	Unique identifier for the loan.
Platform *text*	MPL name.
ProgramType *text*	MPL defined program to which the loan belongs Same as the program fields sent to Arix.

MarketingChannel *text*	Marketing channel name and code as defined by the credit policy.
CreditPolicyVersion *alphanumeric*	Credit policy version.
CreditModelVersion *alphanumeric*	Credit model version.
InvestorName *text*	Name of the investor that decisioned this loan, if applicable.
InvestorDecisioned 1 or 0	1 if the investor decision overrode the MPL decision, otherwise 0.
LoanPurpose *text*	Borrower-stated loan purpose.
AssetClass *integer*	1 - Student 2 - Solar 3 - Debt consolidation 4 - Home improvement 5 - POS 6 - Medical 7 - Auto loan 99 - Other
FinalActionTaken *text*	Decisioning: **Approved**, **Denied**, **Incomplete**, or **Counteroffer** - **Approved** - the application was approved, the offer was accepted and approved by CRB, or it hasn't yet passed the offer stage. - **Denied** - if the application is declined at credit checks. - **Counteroffer** - if a counteroffer is sent to the borrower. - **Counteroffer accepted** - if a counteroffer was sent to a borrower and the borrower accepted. - **Counteroffer not accepted** - if a counteroffer was sent to a borrower and the borrower declined. - **Approved not accepted** - if the application was approved but the offer wasn't accepted. - **CRB denied** - if the offer was accepted and then rejected by CRB.

	- **Incomplete** - if the borrower didn't complete the application.
UnderwriterNotes *text*	Free text field for underwriter notes.
DenialReason *json blob*	json blob of all denial reasons. The blob must EXCLUDE text delimiters and line feeds. ""DenialReasons"": { ""DenialReason1"": ""denial_reason_1"", ""DenialReason2"": ""denial_reason_2"" }" **See note below for explanation**
PolicyDeclineReason *json blob*	Decline reason per credit policy (DTI, FICO, etc.) json blob of all policy decline reasons. The blob must EXCLUDE text delimiters and line feeds. ""PolicyDeclineReasons"": { ""PolicyDeclineReason1"": ""policy_decline_reason_1"", ""PolicyDeclineReason2"": ""policy_decline_reason_2"" }" **See note below for explanation.**
DenialDate *date*	Date the application was denied.
Application Date *date*	Date the consumer applied for the loan.
Reg B Decision Date *date*	Date decisioning occurred.
AdverseActionNoticeDate *date*	Date the MPL emailed/mailed the adverse action notice to the applicant. **Note:** All adverse action notices must be provided to a consumer within 30 days of a denial of credit.

CounterofferDate *date*	Date the MPL provided a consumer with a counter offer in response to their credit application. **Note**: Counter offers must be presented to an applicant within 30 days of receiving a complete application.
NoticeOfIncompleteness *date*	Date that a notice of incomplete application was sent to the borrower.
InitialFraudAlert Y or N	An initial fraud alert can be placed on a consumer's credit report if they think they were a victim of fraud or identity theft. The alert is found on the credit report and stays on file for 90 days. Yes for an initial fraud alert, otherwise no.
ExtendedFraudAlert Y or N	An extended fraud alert can be placed on a consumer's credit report if they were a victim of identity theft. The alert is found on the credit report and stays on file for 7 years. Yes for an extended fraud alert, otherwise no.
AddressDiscrepancyFlag Y or N	This alert, found on the credit report, indicates that the address provided by a user to obtain a report **substantially differs** from the address the NCRA has in the consumer's file. Yes if there is an address discrepancy, otherwise no.
CreditFreeze Y or N	This alert, found on the credit report, restricts access to the credit report. Yes to restrict access, otherwise no.
ExceptionFlag Y or N	Indicates if there are any exceptions to the policy. Yes if there are exceptions, otherwise no.
ExceptionNotes *text*	Description of the exception detailing why an exception was granted.
BorrowerLastName *text*	Borrower's last name.
BorrowerFirstName *text*	Borrower's first name.
BorrowerDOB *alphanumeric*	Borrower's date of birth.

BorrowerSSN *alphanumeric*	Borrower's Social Security Number.
BorrowerAddress *text*	Borrower's street address. Must be a physical address and not a PO Box.
BorrowerCity *text*	Borrower's city of residence.
BorrowerState *text*	Borrower's State of residence.
BorrowerZip *alphanumeric*	Borrower's 5-digit zip code.
BorrowerPhone *alphanumeric*	Borrower's phone number.
BorrowerEmail *alphanumeric*	Borrower's email address.
RequestedLoanAmount *number to 2 decimals*	Amount the borrower requested at the time of application. The requested amount can change before the final loan amount is settled, for borrower or lender related reasons.
LoanAmount *number to 2 decimals*	Actual gross amount borrowed.
OriginationFee *number to 2 decimals*	Charge to the borrower for the loan per the guidelines.
PeriodicPmt *number to 2 decimals*	Amount of regular payment per the note.
FinalPaymentDate *date*	Date of the last regular payment.
FinalPayment *number to 2 decimals*	Amount of the final payment.
FinanceCharge *number to 2 decimals*	Finance charge paid by the borrower.
TotalPayments *number to 2 decimals*	Total paid by the borrower after all payments are made.

PPY *integer*	Number of payments per year.
Rate *number to 2 decimals*	Interest rate, as a percent.
APR *number to 2 decimals*	Annual percentage rate, as a percent.
Amortization *integer*	Number of payments at origination to pay off the loan.
Term *integer*	Number of months at origination to pay off the loan.
RateType *integer* - 1 or 2	1 - Fixed 2 - Floating
PriorLoanFlag Y or N	Has the borrower had other loans with the sponsor. Yes if the borrower has had other loans with the sponsor, otherwise no.
FICO *integer - 3 digits*	Use the 3-digit FICO score only.
FICODate *date*	Date FICO score was obtained.
VantageScore *text*	Vantage score at application.
VantageScoreDate *date*	Date Vantage Score was obtained.
CreditGrade *alphanumeric*	If applicable, internal grade for the loan from the credit policy (Risk Group).
DebtUtilization *number to 2 decimals*	Percent of credit used by the borrower.
TotalRevolvingDebt *number to 2 decimals*	Dollar amount of all revolving debt on the credit report.
CreditInquiries12months *integer*	Number of credit report inquiries in the last 12 months.

DQPast24Months *integer*	Number of times the borrower submitted late payments in the past 24 months on all accounts, per the credit report.
AccountsOpenedPast24Months *integer*	Number of accounts opened by the borrower in the past 24 months, per the credit report.
OpenCreditLines *integer*	Number of open trade lines on the credit report.
CollectionsExcludingMedical *integer*	Number of open collection accounts per the credit report, not including medical.
PublicRecordsOnFile *number of negative public records*	Number of open public records shown on the credit report such as bankruptcies, tax liens and judgements.
MonthsSinceLastRecord *integer*	Number of months since the borrower's last public record.
HomeownerFlag Y or N	If the borrower is already a homeowner - use multiple sources to determine, including the application. Yes if the borrower is a homeowner, otherwise no.
EmploymentLength *integer*	Number of months the borrower has been employed at their current job.
EmployementLengthDescription *text*	Number of years the borrower has been employed at their current job.
Employer *text*	Employer's name.
AnnualIncome *number up to 2 decimals*	Dollar amount of annual income.
NetDisposableIncome *number up to 2 decimals*	Net income remaining after debt (as defined by credit policy, if applicable).
PreDTIExMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI excluding mortgage pre-loan **as a percentage**.
PreDTIInMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI including mortgage pre-loan **as a percentage**.
PreDTIDate *number up to 2 decimals*	**Debt to Income Ratio** DTI date pulled pre-loan **as a percentage**.

PostDTIDate *number up to 2 decimals*	**Debt to Income Ratio** DTI date pulled post-loan **as a percentage**.
PostDTIExMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI excluding Mortgage post-loan **as a percentage**.
PostDTIInMortgage *number up to 2 decimals*	**Debt to Income Ratio** DTI including Mortgage post-loan **as a percentage**.
IPAddress xxx.xxx.xxx.xxx	IP address of the computer the borrower applied on.
MLAFLAG Y or N	If the loan is being made to a covered borrower. Yes if the borrower is covered, otherwise No.
MAPR *number up to 2 decimals*	Military APRA standard calculation used by lenders for loans covered by the Department of Defense's Military Lending Act & Regulation Z. MAPR is required if MLAFLAG = Yes For more information, see: https://www.fdic.gov/regulations/compliance/manual/5/V-13.1.pdf
UpdatedFlag 1 or 0	Flag indicates that this record updates a previous credit decision for this loan. 1 - True - Updates a previous credit decision. 0 - False - Doesn't update a previous credit decision.
MerchantName *text*	Name of merchant, if applicable.
MerchantFee *number to 2 decimals*	Amount paid to merchant, if applicable.
CoBorrowerLastName *text*	CoBorrower's last name, if applicable.
CoBorrowerFirstName *text*	CoBorrower's first name, if applicable.
CoBorrowerAddress *alphanumeric*	CoBorrower's address. Must be a physical address and not a PO Box.

CoBorrowerCity *alphanumeric*	CoBorrower's city of residence
CoBorrowerState *text*	CoBorrower's State of residence.
CoBorrowerZip *text*	CoBorrower's zip code.
CoBorrowerDOB *text*	CoBorrower's date of birth.
CoBorrowerSSN *alphanumeric*	CoBorrower's Social Security Number.
CoBorrowerPhone *alphanumeric*	CoBorrower's phone number.
CoBorrowerEmail *alphanumeric*	CoBorrower's email address.
CoBorrowerFICO *numeric*	CoBorrower's FICO score.
ExtraFields *json blob*	json blob allowing for all extra fields. The blob must EXCLUDE text delimiters and line feeds. Any changes/additions to this field must go through change management.

Note: **DenialReason** versus **PolicyDeclineReason**

The difference between DenialReason and PolicyDeclineReason is a conceptual one:

- The DeclineReason should be the one stated in the AAN.
- The PolicyDeclineReason is the specific rule in the underwriting policy that the applicant didn't satisfy.

  Most of the time they match, but there can be instances when the AAN reason is worded differently than the PolicyDeclineReason.

Example: The AAN (DenialReason) stated "Our partner bank can't approve the loan you requested because we couldn't verify your identity based on the information we received". The reason stated in the PolicyDeclineReason however, is that the applicant was found on OFAC.

Webhooks overview

Arix uses webhooks to update you on the status of your loan, via API calls. Webhooks report to your system with real-time notifications when an event happens. You have to register your URL to receive a webhook for each relevant event status report. When that event occurs, a webhook is triggered and reports the updates to your system.

Example

After the partner submits George's complete loan documents to Arix, the loan passes to the status, *DocsComplete*. That triggers a webhook event which is reported to the partner notifying them of the event.

>
>
> ### Webhook report failure
>
> After a webhook is reported the first time, the endpoint doesn't check that it was successfully received, nor does it retry for failures.

For information on registering for and viewing webhooks, see Register for webhook events and View webhook events and updates.

Call	Description
GetEvents	Returns the events available to subscribe to such as: - FundingAttemptComplete - LoanStatusUpdated - RailUpdated - ComplianceFailed
Register	Registers a URL to an event with these parameters: - EventName (choose from GetEvents) - callback URL (the address to receive the webhook)
GetRegistrations	Returns a list of webhook events you are registered to. - RegistrationId - MPLId - EventName - CallbackURL
Unregister	Unregisters a URL from an event. To unregister from a webhook, use RegistrationId. See View registered webhook events to find the ID.

LoanStatusUpdated

```
{
  "EventName": "LoanStatusUpdated",
  "EventData": {
     "Id": 0,
     "LoanId": "5f1a5e2b-52be-4766-8d7b-aa8d005ecce1",
     "Status": 1,
     "TimeStamp": "2019-07-17T01:45:09.5144036"
  }
}
```

FundingAttemptComplete

```
{
  "EventName": "FundingAttemptComplete",
  "EventData": {
     "Id": 0,
     "LoanId": "28f1388f-8110-4c49-8f5d-aa99010c0575",
     "Time": "2019-07-29T13:44:31.3305937",
     "TransactionGroupId": "77dd9024-8889-4ea9-8424-752d8a37f9b4",
     "FundingResult": 0,
     "AmountFunded": 6748
  }
}
```

ComplianceLoanFailed

Note: Arix checks for duplicated loan numbers within the past 24 hours. If a partner sends a loan with a previously existing loan number, Arix moves the loan to ‘ComplianceLoanFailed’. At that stage, the partner can verify whether to approve or reject the loan.

```
{
  "EventName": "ComplianceLoanFailed",
  "EventData": {
     "MPLId": "xxx",
     "LoanId": "...guid...",
     "CreateDate": "date",
     "FailedRulesReasons": [
        {
           "RuleName": "rule name",
           "Rule": "some string",
           "Data": "some string",
           "Result": true,
           "FailedComplianceID": 67
        }
     ]
  }
}
```

RailUpdated

```
{
   "EventName":"RailUpdated",
   "EventData":{
      "Id":0,
      "RailId":"…ID of rail that is received when rail created…",
      "LoanId":"fd331267-6ac2-4818-9d0f-ab2f00747b61",
      "RailType":8,
      "AmountFunded":8500.0,
      "RailTransactionId":"…railid from cos…",
      "FundResult": int(this is enum RailResultType),
      "Message":"Description of rail created if no error returned",
      "ProcessedAt":"2019-12-26T07:05:09Z",
      "TransactionGroupId":"9dde776d-0c04-4bc8-9c9c-e0ad9a83c447"
   }
}
```

Register for webhook events

This call registers you to webhook events.

Request URL
https://.../Webhooks/Register

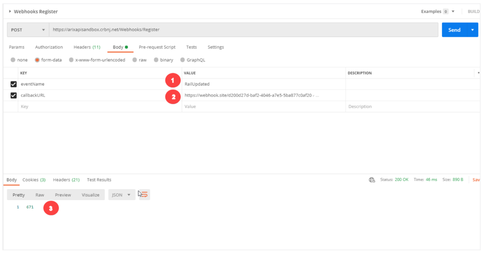
In the request below, you can see that the event registered shows like this, `eventName=RailUpdated`

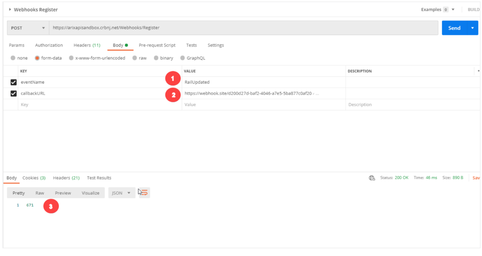
```
curl --location --request POST 'https://arixapisandbox.crbnj.net/Webhooks/Regist
--header 'Accept: application/json' \
--header 'Authorization: Bearer ew0KICAiYWxnIjogIlJTMjU2IiwNCiAgImtpZCI6ICJ6WDU3
--header 'Cookie: ss-opt=temp; ss-id=ENAgtwSchnShvvpW1Is5; ss-pid=syRLWrXbKewzZS
--form 'eventName=RailUpdated' \
--form 'callbackURL=https://webhook.site/d200d27d-baf2-4046-a7e5-5ba877c0af20
'
```

Sample response in JSON - POST /Webhooks/Register

In the image of the response below from the Arix test collection in Postman, the registration number of the new registered event shows.

- #1 shows the `eventName`, `RailUpdated`.
- #2 shows the `callbackURL` which receives the webhook
- #3 shows the registration ID number that was added





Parameter name	Description
EventName *string*	The name of the event to subscribe to.
callbackURL *string*	The URL to receive the webhook report.

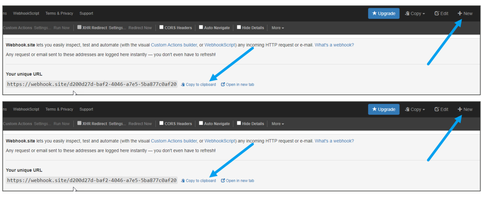
Testing webhook registration

To test registering webhooks you can use the webhook.site and the Arix test collection for Postman. Follow these instructions to test the webhooks.

To register your URL to receive webhook events:

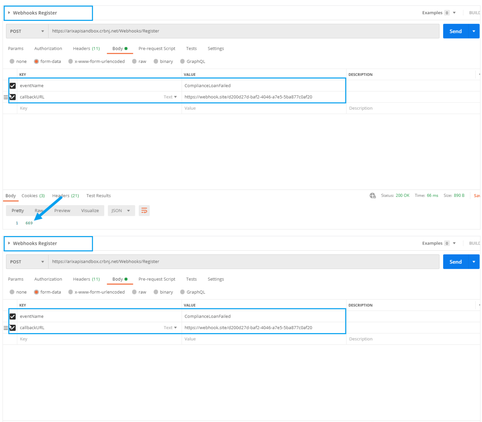
1. Go to https://webhook.site.
2. Click **New** on the top menu.
   The site returns **Your unique URL**.
3. **Copy to clipboard** your **unique URL**.

Note: If you already have other webhooks registered, the browser might automatically direct you to the URL you've used.



4. From the Postman menu, **Webhooks Register**, click the tab, **Params**.
5. In the **KEY** field, enter `eventName` and `callbackURL`.
   For information on event names, click View webhook events available
6. In the **VALUE** fields, enter the event name to register, and the URL you copied.
7. Click **Send**.

The response is the **RegistrationId** of the event. You can find the registration ID with the call, **GET** /Webhooks/GetRegistrations.



Repeat the process for each event that you want to register for.

Note: You can only register for events that you are not currently registered for. If you try to register for an event you are already registered for, or change the callback URL for an event you are already registered for, you will receive an error message in the response body notifying you that you are already registered for the requested event.

Register for webhook events

This call registers you to webhook events.

Request URL
https://.../Webhooks/Register

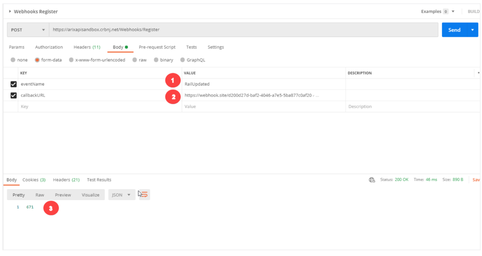
In the request below, you can see that the event registered shows like this, `eventName=RailUpdated`

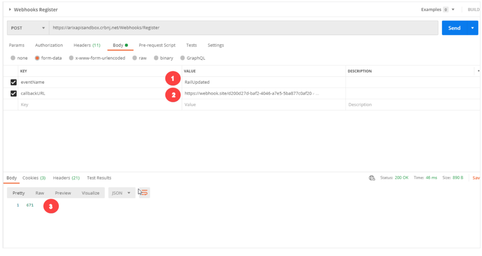
```
curl --location --request POST 'https://arixapisandbox.crbnj.net/Webhooks/Regist
--header 'Accept: application/json' \
--header 'Authorization: Bearer ew0KICAiYWxnIjogIlJTMjU2IiwNCiAgImtpZCI6ICJ6WDU3
--header 'Cookie: ss-opt=temp; ss-id=ENAgtwSchnShvvpW1Is5; ss-pid=syRLWrXbKewzZS
--form 'eventName=RailUpdated' \
--form 'callbackURL=https://webhook.site/d200d27d-baf2-4046-a7e5-5ba877c0af20
'
```

Sample response in JSON - POST /Webhooks/Register

In the image of the response below from the Arix test collection in Postman, the registration number of the new registered event shows.

- #1 shows the `eventName`, `RailUpdated`.
- #2 shows the `callbackURL` which receives the webhook
- #3 shows the registration ID number that was added





Parameter name	Description
EventName *string*	The name of the event to subscribe to.
callbackURL *string*	The URL to receive the webhook report.

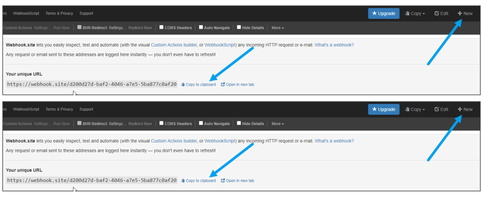
Testing webhook registration

To test registering webhooks you can use the webhook.site and the Arix test collection for Postman. Follow these instructions to test the webhooks.

To register your URL to receive webhook events:

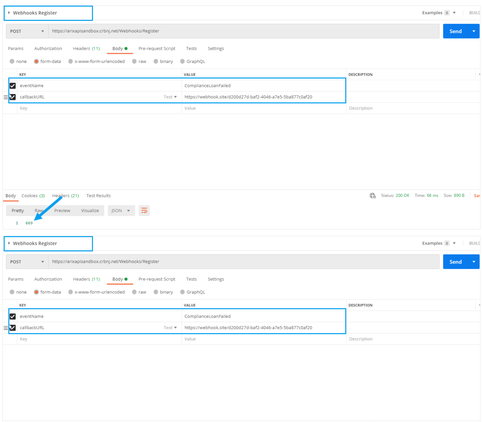
1. Go to https://webhook.site.
2. Click **New** on the top menu.
   The site returns **Your unique URL**.
3. **Copy to clipboard** your **unique URL**.

Note: If you already have other webhooks registered, the browser might automatically direct you to the URL you've used.



4. From the Postman menu, **Webhooks Register**, click the tab, **Params**.
5. In the **KEY** field, enter `eventName` and `callbackURL`.
   For information on event names, click View webhook events available
6. In the **VALUE** fields, enter the event name to register, and the URL you copied.
7. Click **Send**.

The response is the **RegistrationId** of the event. You can find the registration ID with the call, **GET** /Webhooks/GetRegistrations.



Repeat the process for each event that you want to register for.

Note: You can only register for events that you are not currently registered for. If you try to register for an event you are already registered for, or change the callback URL for an event you are already registered for, you will receive an error message in the response body notifying you that you are already registered for the requested event.

View registered webhook events

GET /Webhooks/GetRegistrations

Use this call to view the webhook events you are registered to. These are the webhooks available:

- AdjustmentActionUpdate
- FundingAttemptComplete
- LoanStatusUpdated
- RailUpdated
- ComplianceLoanFailed

For information, see View webhook events available.

For information on registering webhooks, click Register a webhook.

Request URL
https:// ... /Webhooks/GetRegistrations

```
curl --location --request GET 'https://arixapisandbox.crbnj.net/Webhooks/GetRegi
--header 'Accept: application/json' \
--header 'Authorization: Bearer ew0KICAiYWxnIjogIlJTMjU2IiwNCiAgImtpZCI6ICJ6WDU3
--header 'Cookie: ss-opt=temp; ss-id=4R8GJ0cnbN4jS0dU4Dq6; ss-pid=FF4O2hH7WqvHxD
```

Sample response in JSON - GET /Webhooks/GetRegistrations

In the response below, these events are registered:

- **ComplianceLoanFailed**
- **LoanStatusUpdated**
- **FundingAttemptComplete**

Note that each of the `EventNames` have an **Id** number. That is the ID number you will need if you want to unregister a webhook event.


```
[
    {
        "Id": 661,
        "MPLId": "TST",
        "EventName": "ComplianceLoanFailed"
    },
    {
        "Id": 666,
        "MPLId": "TST",
        "EventName": "LoanStatusUpdated",
        "CallbackURL": "https://webhook.site/2acbdfa8-8912-48bb-a58a-9100e141844
    },
    {
        "Id": 667,
        "MPLId": "TST",
        "EventName": "FundingAttemptComplete",
        "CallbackURL": "https://webhook.site/2acbdfa8-8912-48bb-a58a-9100e141844
    }
]
```


Attribute	Description
Id	ID of the event name.
MPLId	ID of the MPL.
EventName	Name of the event that is being reported.
CallbackURL	The destination endpoint that receives the webhook report.

Unregistering webhook events

You can unregister webhook events. For more information, see Unregister a webhook event.

View webhook events available

GET /Webhook/GetEvents

Use this call to view the list of webhook events available for Arix. They are:

- AdjustmentActionUpdate
  This webhook is sent when the details of a loan were updated after the loan was funded.

- FundingAttemptComplete
  This webhook is sent when a funding attempt was completed.
  Important: If you are sending multiple disbursements, wait for this webhook to arrive before you send the next **/FundingInfo** call.

- LoanStatusUpdated
  This webhook is sent when the status of a loan changed. See **Status updates**.

- RailUpdated
  This webhook is sent when a rail has been updated.

- ComplianceLoanFailed
  This webhook is sent when a loan has failed compliance.

Request URL
https:// ... /Webhooks/GetEvents

Sample request in cURL - GET /Webhook/GetEvents

cURL

```
curl --location --request GET 'https://arixapisandbox.crbnj.net/Webhooks/Get
--header 'Accept: application/json' \
--header 'Cookie: ss-opt=temp; ss-id=mEaSEbdHkRyLhcNmkU4n; ss-pid=e3GIciSXVy
```

Sample response in JSON - GET /Webhook/GetEvents

JSON

```
[
    "AdjustmentActionUpdate",
    "FundingAttemptComplete",
    "LoanStatusUpdated",
    "RailUpdated",
    "ComplianceLoanFailed"
]
```

Updated 2 months ago

← View registered webhook events

Unregister webhook events →

Did this page help you? Yes No

Unregister webhook events

DELETE /Webhooks/Unregister

Request URL
https:// .../Webhooks/Unregister

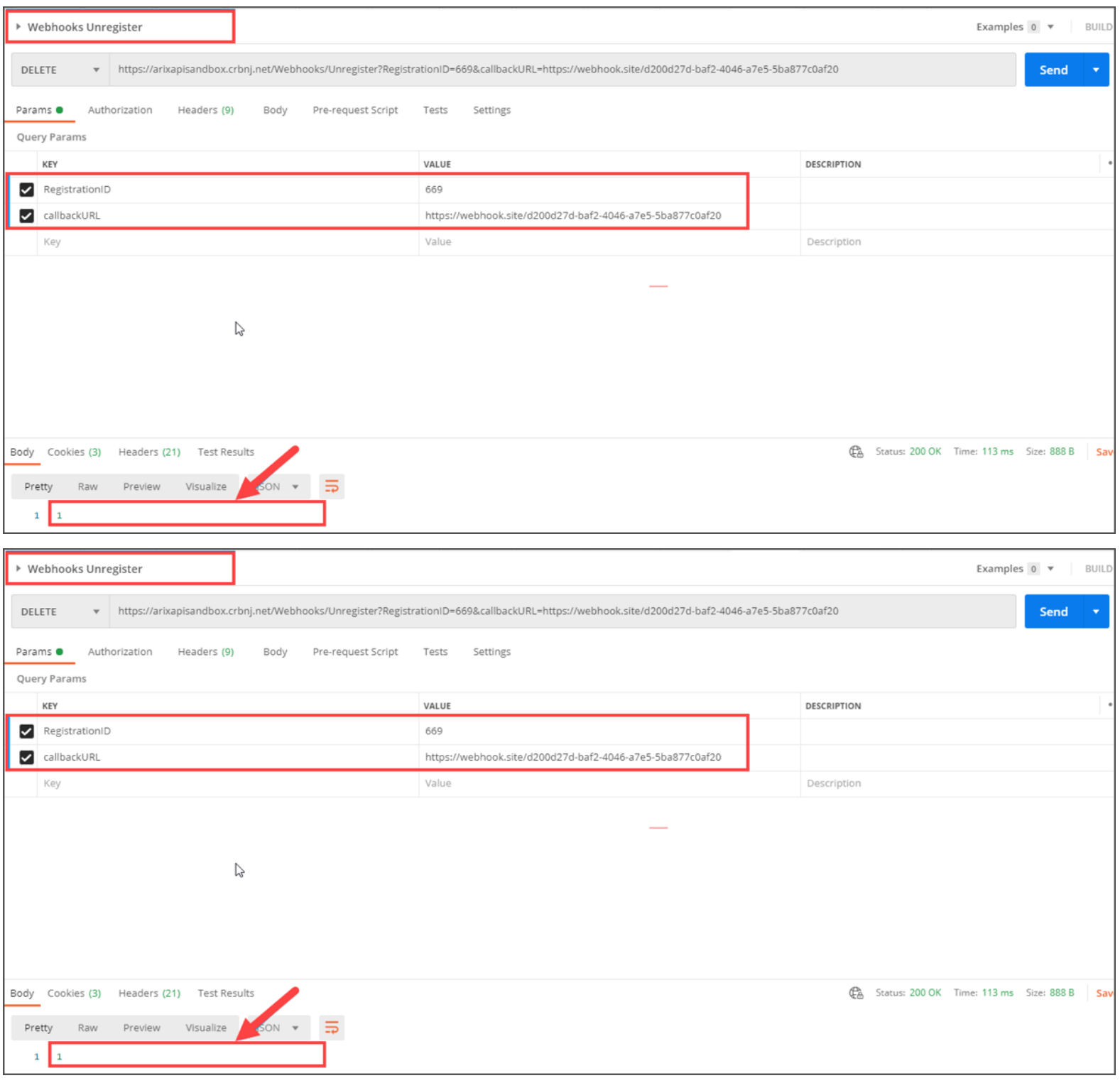
Use this call to unregister any webhook event you are registered to.

```
curl --location --request DELETE 'https://arixapisandbox.crbnj.net/Webhooks/Unre
--header 'Accept: application/json' \
--header 'Authorization: Bearer ew0KICAiYWxnIjogIlJTMjU2IiwNCiAgImtpZCI6ICJ6WDU3
--header 'Cookie: ss-opt=temp; ss-id=QVhFixRMpD92y7HQ7b74; ss-pid=De9QBzZCTdhmCc
```

Parameter name	Parameter type	Data type	Description
RegistrationId	query	long	The registration ID of webhook registration to remove.

Sample response in JSON - DEL /Webhooks/Unregister

There is no response in the image below because the webhook was unregistered successfully.



You can use the Arix test collection in Postman to practice unregistering from an event:

1. In the Postman collection, **Webhooks Unregister** click the tab, **Params**.
2. In the **KEY** field, enter `RegistrationID` and `callbackURL`.
3. In the **VALUE** field, enter the registration ID number that you received when you registered to the event. See View registered webhook events.
4. Click **Send**.

Potential issues (500 series error)

A misconfigured reverse proxy on the client's side can cause GET requests to fail with a 500 series error. Due to the number of possible reverse proxies and ways to configure them, there can be many reasons.

As an example, GET responses return to the originating IP. If this is behind a reverse proxy, or if the configuration is off, a problem can occur with requests attempting to return to a different IP than the originating one.

One successful fix involved the customer changing their TLS connection to specify the server name via SNI, like this:

```
location / {
    proxy_redirect off;
    proxy_ssl_session_reuse off;
    proxy_ssl_name arixapisandbox.crbnj.net;
    proxy_ssl_server_name on;
    proxy_pass_reqest_headers on;
    proxy_set_header Host $host;
    proxy_pass https://crb_arixsandbox_app_server;
    client_max_body_size 50M;
}
```

In this case, `proxy_ssl_name` and `proxy_ssl_server_name` were required parameters for the request to work.

An improved XPay rail

We upgraded the Arix XPay rail to increase reliability. We improved the process and eliminated the 2 step transfer process. We also automated the reversal process - the XPay rail is now simpler and more robust.

Initially, the CRB banking core, COS, allowed XPay transfers from one checking account to another. Arix however, needed a solution to be able to send funds from a General Ledger (GL) account to the partner's checking account.

> 📘
>
> **General Ledger**
>
> A GL is the account type used to create assets (loans) on the banks' books.

To solve that, we introduced a bridge checking account. Every disbursement via the XPay rail consists of 2 legs in COS:

- An internal transfer from the GL to a bridge checking account
- And then an XPay from the bridge checking account to the partner's checking account.

Example of the old 2 leg XPay rail as returned in Get /LoanDetail


```
,,
"ProcessedRails": [
  {
    "Id": 1304627,
    "RailId": "d50718ed-02d6-4dcf-8638-ad5700a400d8",
    "LoanId": "8911dd2b-5bc5-4b91-bf56-ad5700a400d8",
    "RailType": "COSXPay",
    "AmountFunded": 0.000000,
    "RailTransactionId": "5be1eebe-500c-4603-a476-ad5700a40ddb",
    "FundResult": "CosCoreTransactionCompleted",
    "Message": "Test Xpay XPay Transfer From GL to Bridge",
    "ProcessedAt": "/Date(1625047038280-0000)/",
    "DateCreated": "/Date(1625047038238-0000)/",
    "RailAmount": 300.000000
  },
  {
    "Id": 1304628,
    "RailId": "d50718ed-02d6-4dcf-8638-ad5700a400d8",
    "LoanId": "8911dd2b-5bc5-4b91-bf56-ad5700a400d8",
    "RailType": "COSXPay",
    "AmountFunded": 300.000000,
    "RailTransactionId": "27b7eba7-1217-47e7-b630-ad5700a430c9",
    "FundResult": "Success",
    "Message": "Test Xpay",
    "ProcessedAt": "/Date(1625047068240-0000)/",
    "DateCreated": "/Date(1625047038238-0000)/",
    "RailAmount": 300.000000
  }
],
"FundingAttempts": [
,,
"ProcessedRails": [
  {
    "Id": 1304627,
    "RailId": "d50718ed-02d6-4dcf-8638-ad5700a400d8",
    "LoanId": "8911dd2b-5bc5-4b91-bf56-ad5700a400d8",
    "RailType": "COSXPay",
    "AmountFunded": 0.000000,
    "RailTransactionId": "5be1eebe-500c-4603-a476-ad5700a40ddb",
    "FundResult": "CosCoreTransactionCompleted",
    "Message": "Test Xpay XPay Transfer From GL to Bridge",
    "ProcessedAt": "/Date(1625047038280-0000)/",
    "DateCreated": "/Date(1625047038238-0000)/",
    "RailAmount": 300.000000
  },
  {
    "Id": 1304628,
    "RailId": "d50718ed-02d6-4dcf-8638-ad5700a400d8",
    "LoanId": "8911dd2b-5bc5-4b91-bf56-ad5700a400d8",
    "RailType": "COSXPay",
    "AmountFunded": 300.000000,
    "RailTransactionId": "27b7eba7-1217-47e7-b630-ad5700a430c9",
    "FundResult": "Success",
    "Message": "Test Xpay",
    "ProcessedAt": "/Date(1625047068240-0000)/",
    "DateCreated": "/Date(1625047038238-0000)/",
    "RailAmount": 300.000000
  }
],
"FundingAttempts": [
```


This solution worked well for the last year and a half but every once in a while there were some glitches.

Due to the async nature of CRB systems, non-sufficient funds (NSF) in the bridge account could occur at rare times and cause the XPay rails to fail the 2nd leg, as it could not pull the money out.

CRB and the MPLs wasted time identifying the issues and coming up with fixes and the Ops teams would have to make manual transfers and reversals, and support manual data fixes. That's right, manual - for a rail that is expected to work 24/7.

COS now allows direct XPay from GLs to checking accounts and now Arix does as well.

Example of new XPay rail as returned in Get /LoanDetail


```
  "ProcessedRails": [
    {
      "Id": 167938,
      "RailId": "34a65c5e-2e28-498d-bef1-ad6201338e26",
      "LoanId": "87d6da2a-306a-47ff-b3e7-ad6201338e26",
      "RailType": "COSXPay",
      "AmountFunded": 8000.000000,
      "RailTransactionId": "7754aa0f-4aeb-4694-99e6-ad6201339fb7",
      "FundResult": "Success",
      "Message": "A1B2C3",
      "ProcessedAt": "/Date(1626028801897-0000)/",
      "DateCreated": "/Date(1626028801686-0000)/",
      "RailAmount": 8000.000000
    }
  ],
"ProcessedRails": [
  {
    "Id": 167938,
    "RailId": "34a65c5e-2e28-498d-bef1-ad6201338e26",
    "LoanId": "87d6da2a-306a-47ff-b3e7-ad6201338e26",
    "RailType": "COSXPay",
    "AmountFunded": 8000.000000,
    "RailTransactionId": "7754aa0f-4aeb-4694-99e6-ad6201339fb7",
    "FundResult": "Success",
    "Message": "A1B2C3",
    "ProcessedAt": "/Date(1626028801897-0000)/",
    "DateCreated": "/Date(1626028801686-0000)/",
    "RailAmount": 8000.000000
  }
],
```


We also revamped XPay reversals.

If an MPL checking account is configured to allow CRB to pull from it, a return will result in an automatic reversal of the XPay rail. In that case, no manual operation is needed from the Ops team and the reverse is immediate, allowing you to request a new rail without delay.

If an MPL doesn't allow funds to be pulled from their account, the process will stay like it is today. The loan will stay in InReversal until the funds are manually reversed.

If you're an MPL that's already using the XPay rail, your experience will be even better now.

Check's in the mail

Don’t start eulogizing the paper check just yet. Our “Check in the Mail” keeps gaining popularity. This release allows for scalability, security, and a better user experience for our partners.

At the end of 2020 we launched our **Check in the mail** rail. MPLs use checks to give their customers the option of receiving funds or paying off their existing loans via a paper check. Very old school, yes, but our MPL customers jumped at the option.

Our vendor, Checkbook, did a great job of printing and sending out the checks. Because of our architecture, they also managed the check clearing. As the volume increased though, relaying details of cashing, revoking, and expiring checks became a non-scalable procedure.

Positive Pay and COS clearing

To address that issue, we needed to automate the clearing of the checks in-house. COS, our banking core, added a Positive Pay feature, and once Arix integrated that feature, check events could be automatically handled by Arix.

When a check is created, Arix adds it to Positive Pay in COS and it has an expiration date. The check is then printed and mailed out by Checkbook. When a customer tries to cash the check, the fed notifies CRB, and COS reviews the check. The check only gets cashed if all 3 conditions are met:

1. The check is on the Positive Pay list and matches the account information.
2. The check hasn’t expired.
3. The check hasn’t been revoked.

Arix can revoke a check if MPL Operations or the MPL wants to stop a check from being cashed. And, if any of the 3 conditions are not met, COS will bounce the check when someone tries to cash it.

Any event that stops a check from being able to be cashed, such as an expired event or a revoke, will create a return event in Arix (for that check). After a return, you'll be able to request the funds again by check, or by another payment method.

Should I state the obvious? OK, I will.

Since the fed will now send the cash request to CRB, cashed events will now be in real time. No tech changes required by our MPL customers. Relax and enjoy the improved experience.

You may notice these features that we added:

- Check number is now printed in the message field for easier tracking
- Check image is now available to our MPL Ops once it is presented to be cashed

Any failure to create a check will result in a return. This will allow the rest of the loan amount to be disbursed using a different rail.

We are working with Checkbook on other possible mailing options (at present, only USPS).

If you are an MPL that’s keen on disbursing via digital rails, of course we support those as well as payment rails. If a business opportunity arises that requires more traditional payment methods, we’ve got you covered. We’re not burying the paper check just yet.